TECHNISCHE
UNIVERSITÄT
DARMSTADT

CONTEXT-AWARE PHONE MODE ADAPTATION –
Approaches to Classify Phone Position and User Activity from Smartphone Sensor Data

Vom Fachbereich Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

IRINA DIACONIȚA, M.SC.

Geboren am 2. Januar 1988 in Iași, Rumänien

Referent: Prof. Dr.-Ing. Ralf Steinmetz
Korreferent: Prof. Dr. Andreas Mauthe

Darmstadt 2018

BIBLIOGRAPHISCHE INFORMATIONEN

Diaconița, Irina: Context-aware phone mode adaptation – Approaches to Classify Phone Position and User Activity from Smartphone Sensor Data

Darmstadt, Technische Universität Darmstadt

Jahr der Veröffentlichung der Dissertation auf TUprints: 2018
URN: urn:nbn:de:tuda-tuprints-73491
URI: http://tuprints.ulb.tu-darmstadt.de/id/eprint/7349
Version: 20180418_Diaconita_Irina

Tag der Einreichung: 05. Dezember 2017
Tag der mündlichen Prüfung: 26. März 2018

ABSTRACT

Over the past few years both the popularity and the computing and sensing capabilities of smartphones increased significantly. This created tremendous opportunities for pervasive and mobile computing. Previous approaches had to rely on users carrying dedicated devices which gained limited acceptance and spread due to the constraints and extra costs involved. Nowadays a great and ever-increasing number of users own and carry smartphones, thus enabling the monitoring of user context and behavior and the provision of accordingly adapted services while not disturbing the user.

Despite the plethora of features that smartphones provide, a significant proportion of their usage still belongs to communications appplications [9]. In this context, adequate adaptation of the phone notification mechanisms is necessary in order to avoid disruptions and disturbances to the user and surrounding peers, as well as in order to ensure that notifications are not mistakenly overseen. The survey we carried out indicates that a significant percentage of respondents at least occasionally fail to adapt their phone mode, whereas the majority of the participants report having to change their phone modes several times per day.

The goal of this thesis is to provide the users with methods to automatically adapt their phones' notification mechanism based on two types of contextual information that we deemed as essential: the place where the phone is carried and the users' current activities. Phone position detection constitutes our main focus regarding context detection, while for user activity classification we adapt existing approaches to our scenarios. Both types of contextual information are inferred only based on hardware sensors embedded in off-the-shelf smartphones in order to make this approach available to as many users as possible.

The main challenges include obtaining the highest possible classification accuracy and ensuring that the contextual information is not deprecated, given the great risk of disturbance posed by inaccurate results. Furthermore, we aim to limit the duration of sensor sampling in order to reduce both battery impact and privacy concerns.

Our contributions include an audio-based active probing approach which achieves a significant improvement of phone position classification accuracy in comparison to existing state-of-the-art approaches. We subsequently improved this method, which relies on playing audio signals and recording them at the same time, by opportunistically piggybacking phone notification sounds. Thus we could at the same time avoid user disturbance through sound playing, limit the recording duration, and ensure that the classified phone position is actually the current one. For the situations when the phone is on silent mode, we propose an approach that piggybacks vibration motor movements to collect and classify accelerometer readings, achieving accuracies in excess of 90%. Lastly, we provide a proof of concept implementation of a system that awaits incoming phonecalls or notifications, records audio and accelerometer readings, determines the user's activity and phone position, and immediately adapts the phone mode.

KURZFASSUNG

Innerhalb der letzten Jahre hat sich sowohl die Beliebtheit als auch die Rechenleistung und Sensorfähigkeiten von Smartphones erheblich erhöht. Daraus ergeben sich enorme Möglichkeiten für Pervasive und Mobile Computing. Frühere Ansätze mussten sich darauf verlassen, dass Nutzer dedizierte Geräte mitführen – was aufgrund der Einschränkungen und Zusatzkosten zu nur eingeschränkter Akzeptanz und Verbreitung führte. Heutzutage besitzt eine große und steigende Anzahl von Nutzern Smartphones und führt diese dauerhaft mit sich, was eine Beobachtung des Nutzerkontexts und -verhaltens sowie die Bereitstellung entsprechend angepasster Dienste ermöglicht, ohne den Nutzer zu stören.

Trotz der Vielzahl an Features die Smartphones bereitstellen ist ein signifikanter Anteil ihrer Nutzung weiterhin dem Bereich der Kommunikationsanwendungen [9] zuzuordnen. In diesem Umfeld ist die adäquate Anpassung von Benachrichtigungsmechanismen notwendig, um Störungen und Belästigungen für den Nutzer und das Umfeld des Nutzers zu vermeiden sowie um sicherzustellen, dass Benachrichtigungen nicht irrtümlicherweise übersehen werden.

Die von uns durchgeführte Befragung zeigt, dass ein signifikanter Anteil der Befragten zumindest hin und wieder vergisst, ihren Telefonmodus anzupassen, wobei die Mehrzahl der Befragten angibt, diesen mehrmals täglich zu wechseln.

Das Ziel dieser Arbeit ist es Nutzern Methoden zu bieten, mit denen der Benachrichtigungsmodus automatisch angepasst werden kann. Hierbei werden zwei von uns als essenziell eingestufte kontextuelle Informationen analysiert: die Position, an der das Telefon mit sich geführt wird sowie die zurzeit durchgeführten Aktivitäten des Nutzers. Die Ermittlung der Position des Telefons ist der hauptsächliche Fokus unserer Arbeit, während wir uns bei der Klassifikation der Nutzeraktivität auf bereits existierende Ansätze stützen und diese für unser Szenario anpassen. Beide Arten kontextueller Information werden nur von Daten aus Hardwaresensoren abgeleitet, die in gängigen Smartphones verbaut sind, um den Ansatz für möglichst viele Nutzer nutzbar zu gestalten.

Die hauptsächliche Herausforderung liegt darin, die höchstmögliche Genauigkeit der Klassifikation zu erreichen und sicherzustellen, dass die ermittelte kontextuelle Information nicht bereits veraltet ist, da ungenaue Resultate ein hohes Risiko für Störungen mit sich bringen. Weiterhin streben wir an, die Dauer der Sensorbenutzung zu minimieren, um sowohl den Energieverbrauch als auch Datenschutzbedenken zu reduzieren.

Unser Beitrag beinhaltet einen audiobasierten Ansatz mit aktiver Sondierung der eine signifikante Verbesserung der Genauigkeit der Klassifikation der Position des Telefons im Vergleich zu bestehenden State-of-the-Art-Ansätzen mit sich bringt. Anschließend zeigen wir, dass sich in einer Erweiterung dieses Ansatzes eine vergleichbare Genauigkeit erzielen lässt, wenn das Abspielen von vorhandenen Benachrichtigungstönen genutzt wird. Damit können wir sowohl die Störung des Nutzers durch das Abspielen von Geräuschen verhindern, die Aufnahmedauer verkürzen und sicherstellen, dass es sich bei der klassifizierten Telefonposition um die korrekte han-

delt. Für Situationen, bei denen sich das Telefon im Lautlosmodus befindet stellen wir einen Ansatz vor, der die Vibrationsmotorbewegungen benutzt und Daten des Beschleunigungssensors klassifiziert, wobei Genauigkeiten oberhalb von 90 % erreicht werden. Schlussendlich stellen wir eine Proof-of-Concept-Implementierung eines Systems vor, das auf eingehende Anrufe oder Benachrichtigungen wartet, Audio- und Beschleunigungsdaten aufnimmt, die Aktivität des Nutzers sowie die Position des Telefons bestimmt und umgehend den Telefonmodus anpasst.

CONTENTS

1

INTRODUCTION

The advent and increase in popularity of smartphones has created unprecedented opportunities for context-aware approaches. Whereas in the past users had to be persuaded to carry burdensome sensing platforms, nowadays smartphones provide both sensing and computational capabilities and are being willingly carried by the users at most times.

As a consequence, methods to infer increasingly more precise and specific types of context have been developed. These methods span from user localization and path prediction to physical activity recognition and to medical information inference. These new sources of information have been used by a multitude of applications, which could now personalize the content and service for their users without requiring their direct input.

However, a number of challenges persist and some new ones were created. Applications have to be battery-efficient and cannot require much interaction from the user if they aim for a longer term usage. Furthermore, together with advances in context recognition and number and quality of built-in sensors, user privacy concerns have also increased.

1.1 MOTIVATION

Despite the tremendous increase in number and diversity of mobile applications, most used applications still serve for communication purposes. Thus, according to Böhmer et al [9] communication applications account for 49.5% of app launches. Additionally, the average usage of these apps is about one minute. Furthermore, when considering chains of app usage, about half of these chains are started by communications applications, even when the following apps have different purposes.

Since communication stays the main purpose of smartphones, an important issue concerns notification mechanisms. All communication, be it synchronous, like phonecalls, or asynchronous, like emails and text messages,still involves the use of notifications. However the varying daily circumstances of the users require different notification behaviors throughout the day. Thus, a typical user would go from the morning when notifications take their regular form, to commuting where they need to get louder if taking public transport or to potentially be snoozed if the user is driving (particularly text notifications), to work situations when again notifications might get back to regular mode, to meetings where notifications have to be on silent and so on for the rest of the day. One can already notice the number of times the user is required to adapt this notification mechanism, which for simplicity purposes we shall call "phone mode" in what follows. In Chapter 5 we define this term more precisely.

Thus, adapting phone mode is a repetitive task the user has to remember to carry out multiple times a day. Unsurprisingly, sometimes users forget about this task,

which results in disturbances and missed notifications. Thus, while it might be tempting to decide to set the phone to silent mode for most of the day, this can also lead to missing important messages. More importantly, the situations when disturbances are least welcome involve multiple people, such as meetings and presentations, so one user failing to adjust his phone mode results in disturbance for many other people.

There are apps that automatically adapt the phone mode, which rely however on general factors that give only hints regarding the user's situation. For instance, some applications use the WiFi network the user is connected to in order to infer whether he is home, at work, or somewhere in the city, and thus adapt the phone mode. However there are times at work when the user would want to silence the notifications, but others when he would still like to be made aware of them. Similarly, the knowledge that the user is in the city does not suffice. The city includes both loud squares and shopping malls, where notifications should increase their volume, and cinemas and restaurants, when the phone should be on silent mode.

Therefore, looking at more specific contextual information can help make better decisions with regards to the desired phone mode. Ideally, one would like to emulate the user's behavior when deciding the phone mode. Thus, the current activity is one important indicator for the desired phone mode. The phone's position in relation to the user also plays an important role. A very loud ringtone being played while the phone is in the user's hands can be bothersome, while low volume notifications risk not being heard when the phone is in a backpack. Additionally, a phone vibrating on a desk during a meeting is a source of disturbance.

In order to verify the afore-mentioned aspects and obtain more insights into the users' behavior with regards to phone mode adaptation, we carried out a user study which we present in detail in Chapter 4. The main findings included the fact that the majority of respondents adapt their phone modes multiple times per day and are being disturbed at least once per week by another user's phone, while more than a third of the respondents cause a disturbance at least once per week. Most importantly, more than 70% of the survey participants would be willing to use an application that automatically adjusts their phone mode, provided it is implemented in a privacy-conscious manner.

To sum up, phone mode adaptation is a repetitive task that users have to manually carry out numerous times per day. Additionally, forgetting to adapt the phone mode can result in disturbances and missed notifications. Automatic phone mode adaptation would solve this problems. However, existing automated approaches rely only on general indicators of the user's situation, e.g., the WiFi network to which the user is connected, rather than taking into account more fine-grained contextual information, e.g., the phone's position or his current activity.

1.2 GOALS

The goal of this thesis is to provide a solution for automatic phone mode adaptation which allows users to avoid both disturbances and unwanted missed phonecalls and notifications. In order to make our solution available to as many users as possible, we rely solely on the hardware sensors built in smartphones and consider no external sensors. Furthermore, we shall use no software sensors such as calendars, since this would limit our solution to users who regularly use those applications. We addition-

ally dismiss smartphone sensors which are available only on a very limited number of phone models, such as temperature or humidity sensors.

An essential goal concerns the classification quality of the user's contextual information. The main criteria we aim to use for phone mode adaptation are the phone position and user activity. Since user activity detection methods have already been developed [123], we adapt them to our context and proceed to focus on refining methods for phone position detection. Given the high potential of disturbance that an incorrect phone mode presents, it is difficult to deem any accuracy under 100% as good enough. At the very least one could accept errors for situations where the phone should not be in a silent mode. However, since such an accuracy is impossible to achieve, we will strive to obtain as high of a classification result as possible, with related work having achieved up to 84%accuracies [87]. In a real-world implementation, consequences of misclassifications can be partially mitigated by choosing to set the phone on a silent mode in case of low confidence classification results.

We additionally aim to ensure that when a phonecall or notification comes in, our knowledge of contextual information is not deprecated, i.e. the user has moved his phone to a different position or has transitioned to another activity. At the same time, we intend to limit the duration and frequency of recordings as much as possible, both due to battery and privacy concerns. Thus, the former aim discards the usage of duty cycling, while the latter dismisses continuous probing of the sensors. Our goal will be therefore to collect recordings in an opportunistic fashion when the phone mode actually needs to be adapted, i.e. right when a notification comes in.

Classification accuracy should not be achieved at the expense of user comfort and battery lifetime. For this reason, we shall not impose restrictions on the user regarding how he carries his phone or his his environment should be (e.g., not very noisy).

While the current work does not focus on user privacy, we shall take some steps to reduce such concerns. Besides the already mentioned limitation of duration and frequency of recordings, we shall investigate alternative solutions that rely on sensors capturing less sensitive data, e.g. attempting accelerometer-based approaches instead of audio-based ones.

1.3 CONTRIBUTIONS

The main contributions of this thesis are:

- A user study to assess the need for automatic phone mode adaptation and the users' acceptance of such a solution.
- A method to improve phone position classification by using audio pilot sequences, which achieves accuracies of over 90%.
- An approach to opportunistically obtain the pilot sequences by piggybacking all phone notification sounds. This additionally allows us to run the classification only when needed, thus saving battery, and to be certain of knowing the most recent contextual information.
- An active probing approach to determine phone position, which gathers accelerometer readings while the vibration motor is triggered. This method also achieves accuracies in excess of 90% while being suitable for more privacy-aware users.

- Adaptation of existing user activity classification methods to specific user groups, e.g. car technicians.
- A proof of concept implementation of an automatic phone mode adaptation system.

1.4 OUTLINE

The remainder of this thesis is structured as follows. Chapter 2 introduces the main concepts that will be used in this thesis, which include machine learning concepts and the statistical methods used in our motivating user study.

Subsequently, we summarize the main related work in Chapter 3 and highlight remaining open questions and challenges which will be targeted in our work. Chapter 4 presents our user study concerning frequency of manual phone mode adaptation and disturbances originating from failing to adapt it. Additionally, it investigates several issues which affect our approach, such as the most common phone positions and activities, as well as the preferred way of being supported with phone mode adaptation, e.g. automatic adaptation vs. recommendations. Chapter 5 presents our approaches for phone position classification, introducing machine leaning techniques and active probing methods we used. It starts with the more basic approaches and progresses based on remaining challenges towards the more user-friendly and battery-saving ones used in our final proof of concept. Each method presentation is paired with an evaluation section. Chapter 6 introduces two sets of user work-related activities we considered, one pertaining to car technicians and the other to office workers. The former one has been implemented and evaluated, while the latter one is implemented in the proof of concept application without an evaluation section. Subsequently, we introduce the main architecture and implementation concerns together with the decisions taken for the proof of concept application, which we present in the final part of the chapter. Finally, Chapter 7 summarizes our contributions and compares them with the goals previously outlined in the current chapter.

2

FUNDAMENTALS

In the following chapter we introduce the main concepts used in this thesis. We first define context and its components. We proceed to introduce the main steps of sensor-based context classification. We present the basic concepts of machine learning, focusing afterwards on supervised learning and classifier performance metrics. Lastly, we present the statistical tests used in our user study.

2.1 CATEGORIES OF CONTEXTUAL INFORMATION

One way to gain information about the user is the direct input from the user himself. However, this approach is often burdensome to the user, who has to input every single aspect of his status. This is especially true given the size of most smartphone screens. In turn, this limits application developers with regards to the amount of information they obtain from the users. Additionally, the application has to wait for new input form the user and cannot predict his next actions and decisions. The last years have seen a great increase in applications that go beyond this, inferring as much as they can about the user's situation.This is what is often called, both informally and in research papers, context.

Before the advent of context, linguistics has introduced and made use of the term "situation", defined by Barwise [6] as:

"The world consists not just of objects, or of objects, properties and relations, but of objects having properties and standing in relations to one another. And there are parts of the world, clearly recognized (although not precisely individuated) in common sense and human language. These parts of the world are called situations. Events and episodes are situations in time, scenes are visually perceived situations, changes are sequences of situations, and facts are situations enriched (or polluted) by language."

In the mobile computing area, one of the most comprehensive, widely-accepted and used definitions belongs to Dey [24], who describes context as follows:

"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves."

Earlier definitions included "a situation and the environment a device or user is in" [130]. Context-awareness had been previously defined by Schilit et al. [128] as "the ability of a mobile user's applications to discover and react to changes in the environment they are situated in", while Schmidt et. al [129] define it as "knowledge about the user's and IT device's state, including surroundings, situation, and, to a lesser extent, location." Each of these definitions misses certain types or aspects of context and for this reason, Dey's definition stands as the most comprehensive one.

Thus, context includes three essential aspects or, as we might call them, subcontexts: the user, the application together with the device it runs on, and the in-

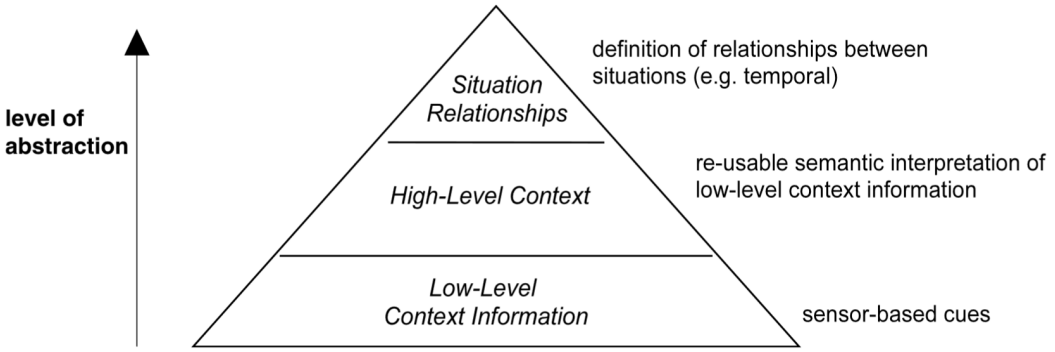


Figure 1: Layers of context as categorized by [7]

teraction between them. Worth noting is that context refers to both the individual pieces of information and the whole picture these pieces create. Since the definition of context is very broad, the practical definition of context will differ between applications based on what they deems as essential information for their use case. While there is no single framework exhaustively defining and structuring context and thus making a complete context taxonomy there are proposals for guidelines in modelling context [7, 46]. Due to the inevitable ambiguity and generality, several applications use "context" to refer to location, ambient, mood or sentiment, (physical) activity, health status etc.

Context can be split into layers and categories, based on the granularity and complexity of the information it contains, as well as on the entities it concerns (e.g., user, device etc.). In this regard as well authors' opinions differ. In what follows we present two approaches to context components. Bettini et al. [7] split context on three layers: low, high, and relationships, as also shown by Figure 1. The low layer concerns sensor readings themselves, while high-level context concerns semantic information extracted based on these readings, regardless of its granularity. Sitting down or eating lunch would both qualify as high level context. The highest level of their model, which they do not even call context anymore, refers to relationships between contexts, e.g., the temporal relationship between standing up and walking. Schmidt et al. [129] split context based on the element it concerns, namely self, environment and activity, as shown by Figure 2. The self category concerns both the user and his device, the environment refers to both the physical and social setting, and activity to the task the user is carrying out.

Applications using contextual information can take multiple names, such as context-aware or adaptive applications. According to Steinmetz and Nahrstedt [144], "adaptive systems monitor the user's activity pattern and automatically adjust the interface or content provided by the system to accommodate (...) changes in user skills, knowledge and preferences."

In the current thesis, we target two specific aspects of context: user activity and phone position. In the existing body of work, user activity most often refers to physical activity [5, 18, 19, 63]. However this is an inaccurate representation, since physical activities are only a subset of human activities. Additionally, the afore-mentioned issues concerning the definition of context layers affect the definition of user activity as well. Depending on the level of granularity considered, typing, writing a report, and working on the MOLEM project can all represent an activity, even the very same

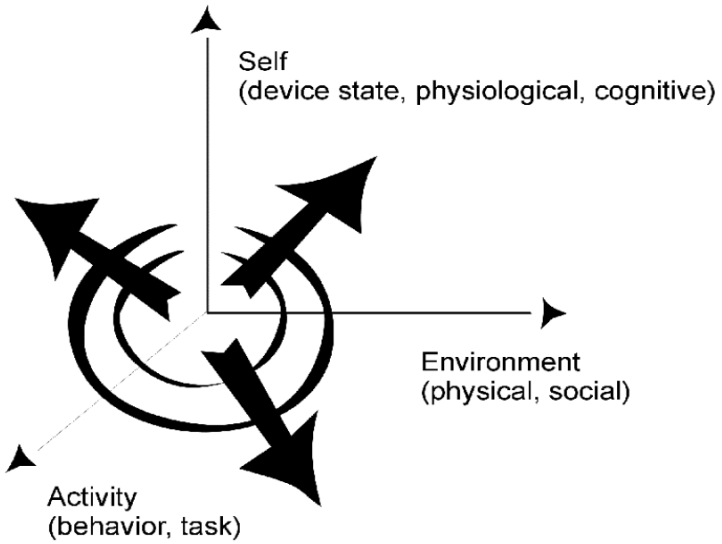


Figure 2: Context sub-categories according to [129]

one. For this reason, developers of applications have to decide on the meaningful level of granularity together with the adequate class set.

Phone position refers to the place where the user carries or stores the phone, having no connection to the actual geographical location. Phone positions would include, for instance, being inside a pocket or on a table. One can also consider different levels of granularity here, from considering only if a phone is inside or outside of an enclosure, to taking into account whether the pocket is a shirt or jacket pocket or whether the table is made out of wood, metal, or plastic.

2.2 SENSOR DATA-BASED CONTEXT DETECTION USING MACHINE LEARNING

Our work, as well as most of the recent related research leverages machine learning techniques. According to Mitchell [91], machine learning can be defined as follows: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Machine learning includes four main types of algorithms: supervised, unsupervised, semi-supervised, and reinforcement learning algorithms, as shown in Figure 3. Supervised algorithms build models based on existing labeled data so that when presented with new unlabeled data, they can predict their label. For instance, we provide our supervised learner with a set of accelerometer samples and labels corresponding to the user's physical activity, it builds a model based on that and afterwards, when provided with new unlabeled accelerometer readings, it can classify the user's activity. Unsupervised methods aim to infer a function from unlabeled data, relying on discovering hidden structures in the data. Thus, they can find patterns and clusters in data but cannot assign a value or label. Unsupervised models are often used to find anomalies and anomalous behaviors in the data, e.g. [137]. In the previous example, an unsupervised model would find different clusters corresponding to the various physical activities, but would be unable to name the activities. In the case of semi-supervised learning, the algorithm is fed both labeled and unlabeled samples,

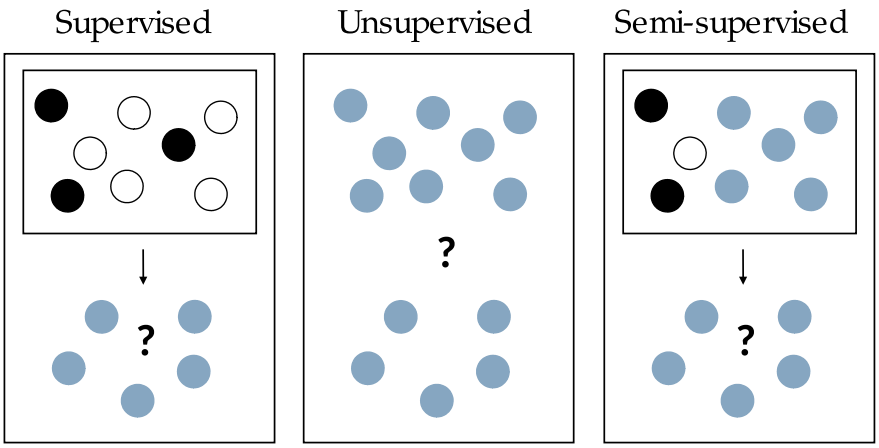


Figure 3: The main types of learning algorithms

most often the latter being more numerous than the former. Similarly to the unsupervised case, such a learner first identifies patterns and clusters in the data. Afterwards, using the available samples, it tries to assign labels to these clusters. Reinforcement learning targets the learning problem from a different perspective. Thus, instead of focusing on correct or incorrect labels, it rewards desired behaviors and penalizes unwanted behaviors by using a reward function. In the process of optimizing the reward function results, these algorithms also learn a model of the data without being provided explicit examples. For instance, if a user has a playlist that he uses only when he goes jogging, a classifier could learn to recognize the movement patterns associated to jogging just from starting the playlist every time it think the user is jogging, observing whether the user changes the playlist and adapting its model accordingly.

In the current work we rely only on supervised algorithms. Formally put, supervised learning represents [124]:
"Given a training set of N example input-output pairs $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$ where each y_i was generated by an unknown function $y = f(x_i)$, discover a function h that approximates the true function f."

Supervised learning algorithms require a training phase, when they are provided with labeled data instances, based on which they learn a model. Afterwards, the model can be used to label unseen data points.

Two of the most common machine learning tasks are classification and regression. Thus, if the predicted output is discrete, we are presented with a classification problem. Regression problems have a continuous output. Our work concerns itself only with classification problems. A typical classification pipeline consists of data acquisition, data cleaning, feature extraction, and running the chosen machine learning algorithm. Adapting this to our work, context detection consists of four main steps: sensor data acquisition, preprocessing, feature extraction, and classification. There are significant variations within each step, however almost all related work (a few examples being [2, 86, 97]) agrees on the steps themselves.

2.2.1 *Data Acquisition*

In the current work we refer only to data gathered from smartphone sensors with the aid of mobile applications. There are two ways to approach data collection. The simplest method is to just sample the smartphone sensors which would be subsequently fed into the classification pipeline. This method can also be called passive probing. Opposed to it is the method of active probing, which does not simply collect the sensors readings, i.e., but also at the same time triggers a pilot sequence. In telecommunications, pilot sequences are signals "transmitted over a communications system for supervisory, control, equalization, continuity, synchronization, or reference purposes" [39]. In mobile sensing pilot sequences are most often used as ground truth for reference purposes [14], however other applications such as supervision are represented as well [20].

There are several approaches to sensor sampling as well: continuous sampling, duty cycling, and opportunistic sampling. In the first case, as the name suggests, data is sampled continuously and then fed into the classification pipeline. This approach presents significant disadvantages such as rapid battery draining and increased privacy concerns, however it ensures that the application is provided with the user's most recent context (as provided by the chosen model, with its respective classification accuracy). Depending on the phone model and operating system, continuous sampling may be disabled. Duty cycling refers to samples being collected periodically for given time durations. Thus, it involves a trade-off between context actuality on the one hand and battery lifetime and privacy on the other hand. Several mobile sensing approaches include improving this trade-off in their focus [86]. Opportunistic sampling refers to piggybacking a different event in order to get the data, such as using photos that the user took for Instagram, or the accelerometer data he recorded for the running app. Such an approach has the advantage of getting the data free of any additional "cost", however it can also involve getting insufficient data or not having the data at the moment it is needed if the piggybacked events are not correlated with the events one is trying to detect.

2.2.2 *Data Preprocessing*

The very first step, once the data has been acquired, is data cleaning. This process depends on the type of data and the data collection process, and deals amongst others with incorrect or missing entries in the dataset. There are multiple ways of dealing with such issues, from removing the whole input vector if one element is obviously wrong or missing, to replacing this element with, for instance, the average value of all corresponding vector elements in the dataset. To make this process more understandable, we shall provide two data cleaning examples, one for gathering training data and one for the actual classification.

Let us first assume we gather training data from all sensors incorporated in a smartphone for a user activity classification task. The user decides when to start and stop the data collection and labels the collected data with the corresponding activity. First of all, we shall discard the first and last few seconds of every recording since the user needs a small amount of time to actually start with the activity after inputting it, and similarly finishes it before inputting it to the phone. Additionally, recordings

over a certain length will be removed assuming the user forgot about the recording process.

For the data cleaning step for an actual classification task, where a trained model is present, let us consider an application that plays a very short audio track, records it using the very same phone and the classifies the phone position. In this case we must check in real time whether our recording is clipped due to the sound volume, so if needed we can adjust the volume and make another recording. Additionally, given the duration of the played sound, we need to start the recording before playing starts. This is a precaution given the time variations of the mobile operating system event scheduling. Therefore, the first few milliseconds of recording will have to be dismissed.

In what follows we present the standard preprocessing steps for audio data. For the other phone sensors the most common preprocessing steps include, besides the afore-mentioned data cleaning step, only the windowing of the data. We describe windowing what follows for audio data only, the process being similar for the other types of sensors.

2.2.2.1 *Audio Signal Preprocessing*

First of all, the recording needs to be split into multiple windows if the total recording length is longer than the optimal size. Depending on the classification task, there can be an optimal constant window size, or the window size might vary. The first case is usually associated with trying to classify environment characteristics that vary little over time and usually have a long duration. One example when this applies is when classifying the transportation mode of the user: the bus, tram, or train will have their specific acoustic pattern over the whole recording. In this case, the only problem is that long recordings will lead to less accurate classification results. The reason for this is that before the actual classification step we apply a feature extraction algorithm in order to reduce dimension, which in the case of too long recordings will result in averaged values which cannot capture the relevant patterns. The optimal length is often decided empirically for each scenario. The second case, when a variable window size is required, applies when trying to capture a short transient event, for instance a knock on a door or a key being pressed. We shall only consider in what follows the case of constant window size, since that is what our scenarios requires.

In this step, a window function is applied [96]. The recorded audio signal is likely not periodic and even if it is, the segments we split it into will most likely not contain an integer number of periods, since the snippet size was chosen to optimize classification accuracy. Therefore, the endpoints of these signals will be discontinuous, which will appear in the frequency domain representation as high frequency components which did not exist in the original signal. This phenomenon is known as spectral leakage. In order to alleviate this issue, the sequence is multiplied with a finite window whose amplitude varies gradually towards zero at the edges [96]. This results in the signal's endpoints meeting and thus eliminating discontinuities. However, using a window function has also side effects, hence the necessity to select an appropriate window function from the several options available. Each such window has a main lobe and several side lobes and there is a tradeoff between the size of these lobes. A few examples are shown in Figure 4. Lower side lobes reduce spectral

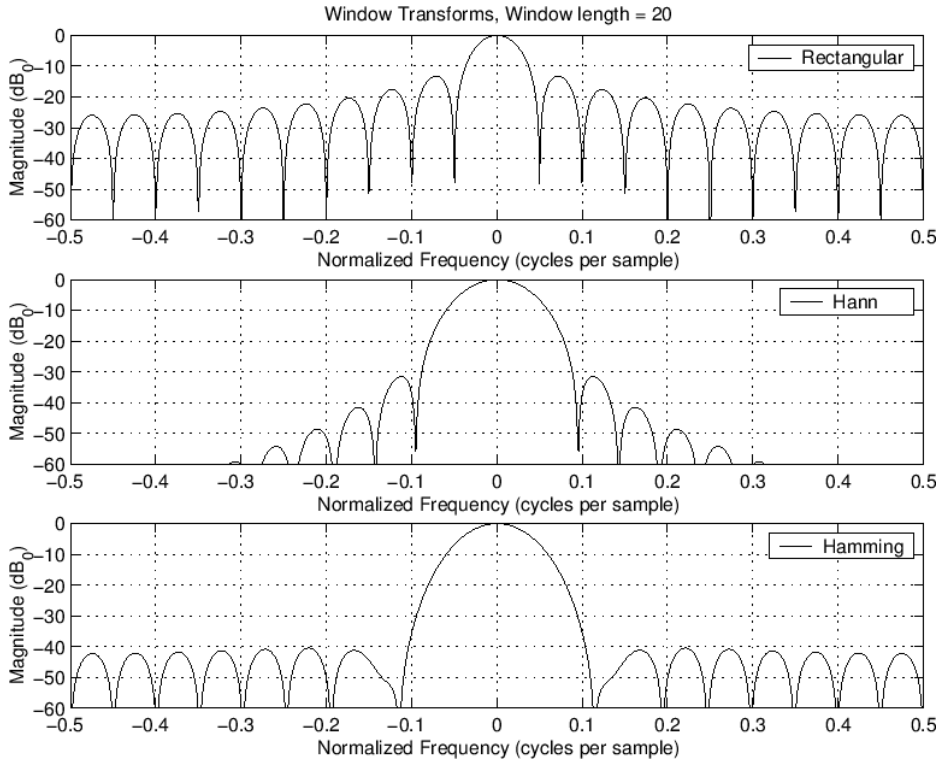


Figure 4: Rectangular, Hann, and Hamming window functions (from [139])

leakage, which is the main goal when using a window function but at the same time increase the bandwidth of the main lobe. On the other hand, higher side lobes are associated with a higher spectral leakage. Out of the numerous window functions, we shall refer in what follows to three of them: rectangular, Hanning (Hann), and Hamming functions, shown in Figure 4. Applying a rectangular function is the same with not applying any window function. Hanning and Hamming functions have both a sinusoidal shape and both have a wide main peak and low side lobes. The Hanning function reaches zero on both sides, effectively removing all discontinuity, while Hamming does still have some discontinuity left.

The second step is silence removal. If left in the dataset, silent windows will be most likely misclassified and harm the overall classification performance. In certain scenarios where it is most unlikely to have silent moments, this step can be skipped, provided that processing time is a critical aspect. Therefore, we compute the signal energy for each window. If it is under a certain experimentally-established threshold, the window is discarded, otherwise it is forwarded to the next step of the pipeline.

In the next stage we transition from time to frequency domain by applying a Discrete Fourier Transform (DFT) to each window. While for a continuous sample the continuous Fourier transform would be used, for signals which are known only at N points equally spaced in time, the DFT can be applied [121]. This representation shows the frequency components of the analysed signal. To put it formally, if $f(t)$ is the continuous signal from which we get our data and $f[0], f[1], ..., f[N-1]$ are N samples from this signal, then the DFT is defined as:

$$F[n] = \sum_{k=0}^{N-1} f[k] e^{-i\frac{2\pi}{N}nk}, \qquad n = 0 : N-1 \tag{1}$$

In our work we computed the DFT using a Fast Fourier Transform (FFT) algorithm. More specifically, we relied on the NumPy [154] Python library which implements the Cooley-Tukey FFT algorithm.

2.2.3 *Feature Extraction*

In this step, a feature extraction algorithm is applied in order to reduce the dimensionality of the data. High dimensional data is not only associated with higher computational costs, but also often leads to poorer classification results due to overfitting. Thus, higher dimensional data is prone to having more noise which leads to more noise being modeled by the learning algorithm, i.e. overfitting. As we explain in more detail in Section 2.2.5, this means that the model has a low generalization capacity, performing very well on the data it was trained on and much worse than non-overfitted models on unseen data. In the following sections we describe a range of feature extraction algorithms commonly used for audio data as well as for data from other sensors.

2.2.3.1 *Features for Audio Data*

A particular attention has been given in the past to speech recognition, which lead to the engineering of various types of features. These have been naturally used also for similar tasks such as speaker recognition, or related topics such as music genre classification. However, these features can be quite successfully used to classify totally different types of sounds, as discovered by numerous approaches aiming to classify various environment sounds. By environment sounds we mean all types and sorts of sounds which were considered of interest in various scenarios, excluding human speech and voice recognition. Human voice can be included in these recordings, but it will not be the main focus but at most a hint for determining an element of interest. For instance, when classifying a meeting or party, there will be human voices in the recording, but they and their "content" is not the focus here. Thus, related literature has used both simple and more sophisticated features for environment sounds classification. Simple features include band energy [33] and power spectrum, but besides the obvious advantages of simplicity and ease to compute, usually yield poor classification results. For this reason, they resort to more sophisticated methods, the most commonly used and one of the most successful being Mel Frequency Cepstral Coefficients (MFCC), together with other features derived from it, such as Delta Mel Frequency Cepstral Coefficients (DMFCC or Delta MFCC) and Delta Delta Mel Frequency Cepstral Coefficients (DDMFCC).

MFCC are short-term spectral-based features which owe their success to being "able to represent the speech amplitude spectrum in a compact form" [73]. They are "perceptually motivated" [150], emulating the behavior of the human auditory system [141]. In what follows we describe the way these features are computed ac-

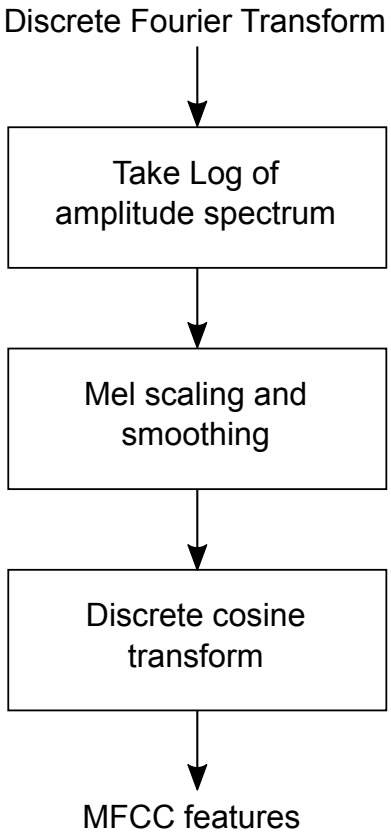


Figure 5: Steps to compute the MFCC features for one frame, according to [73]

cording to [73], while the main steps are shown in Figure 5. Throughout all these steps one should note the importance given to user perception.

The Fourier Transform of the given frame serves as input. In the first step, phase information is discarded on the basis of perceptual studies that have shown that the amplitude of the spectrum is more important than phase. Next the logarithm of the amplitude spectrum is computed, given that the perceived loudness of a signal is approximatively logarithmic. In the second stage the spectrum is smoothed and perceptually relevant features are emphasised. This is done by putting the spectral components in a number of bins, spaced according to the Mel Scale. In the case of human speech – but potentially in other cases too – lower frequencies are more important than higher frequencies from a perceptual point of view. The Mel Scale maps actual frequencies to perceived pitch. Thus, instead of spacing the bins equally based on frequency, we space them equally based on perceived pitch according to the Mel Scale. The vectors that are obtained in this step are highly correlated which allows us to further reduce the dimension of the data in the next step. Thus, a transform is applied to decorrelate the components of these vectors. This could be done by the Karhunen-Loeve Transform (or also the Principal Component Analysis), but in practice is instead approximated by Discrete Cosine Transform. At this point N coefficients are obtained from this frame, most commonly $N = 13$.

2.2.3.2 *Features for the Remainder of Sensor Data*

In what follows we shall refer to some of the more common features for smartphone sensor data. Obviously these feature apply only to sensors with numeric readings and not to binary output sensors such as the proximity sensor, or sensor measurements that are directly attributed a semantic meaning, such as GPS readings. In our work we use this type of feature for accelerometer and light sensor readings, but they can and have been used to classify other sensor readings too, such as barometer, magnetometer, gyroscope, or temperature sensor.

We consider the following features computed across each sample window: maximum, minimum, median, root mean square, Pearson correlation and standard deviation. If the readings have multiple dimensions, the feature function will be applied on each of these dimensions, all resulting values forming the feature vector, as shown for the case of accelerometer readings by Equation 2.

$$\begin{aligned} \textit{features(data)} = (&\min_{x,y,z}(\mathtt{data}), \max_{x,y,z}(\mathtt{data}), \mathtt{median}_{x,y,z}(\mathtt{data}), \\ &\mathtt{rms}_{x,y,z}(\mathtt{data}), \mathtt{pcorrelation}_{x,y,z}(\mathtt{data}), \mathtt{sd}_{x,y,z}(\mathtt{data})) \end{aligned} \tag{2}$$

2.2.4 *Classification*

In what follows we briefly describe the classifiers we experimented with in our work. Some of the considered algorithms can be used both for classification and regression, however in what follows and the overall work we shall refer only to the classification problem.

Naïve Bayes is a set of classifiers which rely on applying Bayes' theorem. It includes Gaussian, Multinomial, and Bernoulli Naïve Bayes, out of which we focused on the first method. The "naïve" assumption is that all features are pairwise independent, as expressed by Equation 3. The difference between the different types of Naïve Bayes classifiers lies in the distribution they assume for $P(x_i|y)$. Based on the afore-mentioned assumption and Bayes' theorem, it is proven that the actual class $\hat{y}$ is given by Equation 4, where $P(y)$ is approximated as the relative frequency of class y in the training set and $P(x_i|y)$ is calculated based on the probability distribution assumed by the specific type of Naïve Bayes classifier used [79].

$$P(x_i|y, x_1, ..., x_i - 1, x_i + 1, ..., x_n) = P(x_i|y) \tag{3}$$

$$\hat{y} = \underset{y}{\mathrm{argmax}} P(y) \prod_{i=1}^{n} P(x_i|y) \tag{4}$$

Despite its simplicity and its name, the Naïve Bayes algorithms perform well in several real-life applications such as spam filtering and document classification, even when provided with small training sets. Given the high bias of the algorithm, i.e. the assumption of independence between features, the algorithms also present low overfitting risk.

Decision Trees are a non-parametric predictive model which can be used for both classification and regression [122]. It relies on decision rules inferred from the training data features [104]. A decision tree is built in a greedy fashion, starting from the root. At each step the new node or criteria to split the data is selected in order to maximize entropy, i.e. minimize Gini impurity or maximize the information gain. In

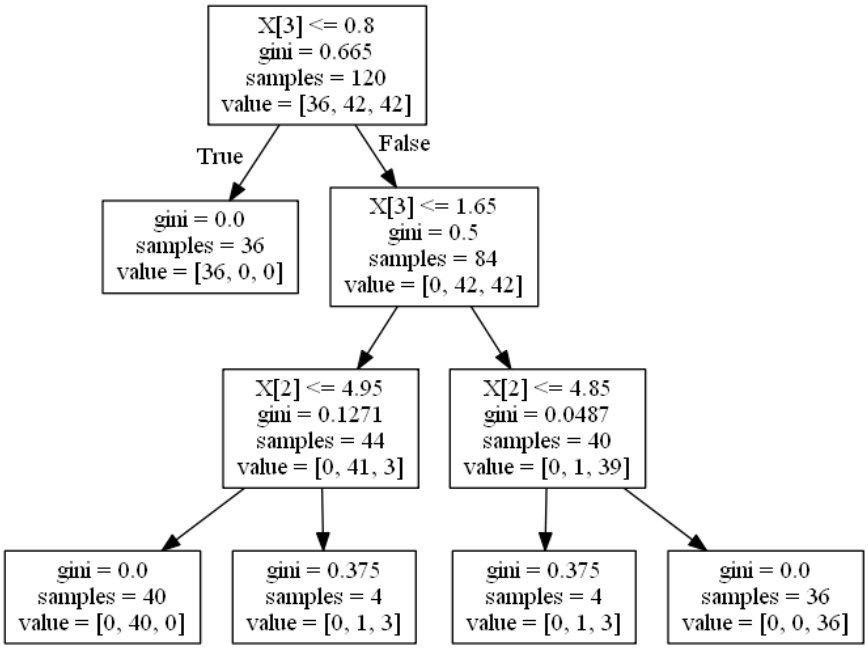


Figure 6: A decision tree trained on the Iris dataset

its most basic form, the process is completed when each leaf has elements belonging to a single class. The prediction process consists of walking the tree starting from the root, each node containing a decision, based on which the following child node is decided. The walk ends when a leaf is reached, which represents the predicted class. One can notice the similarity between this algorithm and the well-known principle of "divide and conquer". Advantages of decision trees include the ease to interpret and be understood even by laypeople. The obvious decision process make decision trees a white box classifier, as opposed to many if not most classifiers which can be regarded as black boxes. On the other hand, decision trees can create biased models if the training dataset is not balanced and are prone to overfitting especially when using large feature vectors [104]. Additional techniques such as pruning can be used to mitigate this latter issue. A sample decision tree built to model the famous Iris dataset [36] is shown in Figure 6.

Random Forest is an ensemble averaging method, which means it aims to improve the generalization ability (and thus reduce overfitting) for a certain algorithm, in this case decision trees, by building several models and then averaging their prediction. Thus, random forest builds several trees from randomly chosen bootstrap samples, i.e. samples drawn with replacement. Additionally, the process of splitting nodes, i.e. selecting the condition that needs to be checked in order to proceed to the next node, does not consider anymore the best split amongst all features but only amongst a random subset of the features [10]. Depending on the implementation, the final class represents either the average of individual tree's votes [10] or the average of probabilistic predictions of each individual tree [104].

Nearest Neighbors are distance-based methods for both supervised and unsupervised learning. The k-Nearest Neighbors classifier works by performing a majority vote on the k closest neighbors to the instance to be classified in the n-dimensional space, where n is the size of the feature vector. Thus, the class with the highest number of instances within the k nearest neighbors is the predicted class of the new

sample. Since the classifier does not build a specific model for the data, but rather simply bases its predictions on distances, the choice of k, the number of neighbors to be considered, is essential.

A Gaussian Mixture Model is "is a parametric probability density function represented as a weighted sum of Gaussian component densities" [120] and is described by Equation 5. Thus, the model assumes all data points to come from a finite number of Gaussian distributions with unknown parameters [104]. Gaussian Mixture Models are often used in speech recognition scenarios given their ability to model "a large class of sample distributions" [120].

$$p(x|\lambda) = \sum_{i=1}^{M} \omega_i g(x|\mu_i, \sum_i) \tag{5}$$

where x is a D-dimensional data vector, ω_i are the mixture weights and $g(x|\mu_i, \sum_i)$ are the component Gaussian densities.

2.2.5 *Classifier Evaluation Issues: Overfitting, Underfitting, Cross Validation*

One of the most important characteristics of a machine learning model is its ability to generalize, i.e. to perform well on unseen data. Thus, two important concepts are those of in-sample error, or training error, and out-of-sample error, or validation error. The in-sample error represents the error we obtain when evaluating the model on the same data we have trained it, and is a measure of how much of the dataset it managed to include in its representation. Out of sample error concerns the classification error obtained when presenting the model with new data.

A high in-sample error indicates an underfitting model, that is too simple to represent the underlying distribution of the data. This issue can be also described as having a high bias. The converse problem, which is overfitting, corresponds to having a low in-sample error and high out-of-sample error. This happens due to the fact that as we increase the complexity of our model, the in-sample error keeps decreasing until we found a function that includes all points in our dataset. However, in this process we have also modeled all noise present in the dataset. This problem is also called having a high variance.

Thus, the in- and out-of-sample errors have to be considered together in order to select the optimal complexity of a model, i.e. its hyperparameters. Thus, as we start with the simplest version of the model, both in- and out-of-sample errors will be high and will keep decreasing as we increase the complexity of the model. However at a certain point the out-of-sample error will reach its minimum and further increasing the complexity of the model will only lead to the increasing of out-of-sample error, despite the in-sample error continuing its downward trend. This process is also represented in Figure 7, belonging to [78].

Since labeled data is often scarce, it is not feasible to use different datasets for training and validating the model. Thus, the same dataset must be split and used for both tasks. A typical method employed in this case is cross-validation [132]. Its simplest form is the holdout method, where a part of the data is used for training and another one for testing. This method however depends heavily on the distribution of data in the training and testing set, giving evaluation results a high variance. A better approach is provided by the popular n-fold cross validation, which involves

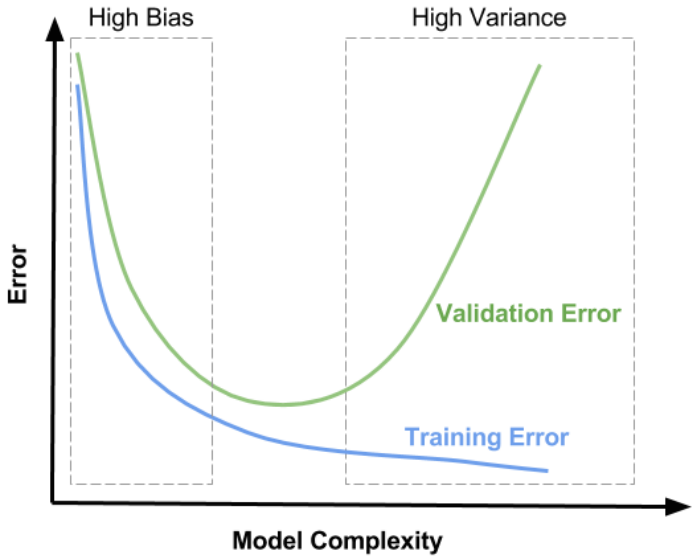


Figure 7: The bias variance trade-off (from [78])

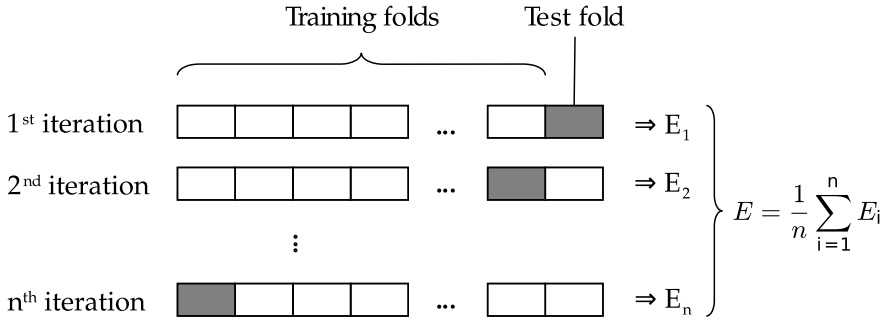


Figure 8: The n-fold cross validation method

splitting the data in n folds and carrying out n iterations. As shown in Figure 8, at each iteration $n-1$ folds are used for training and one of them for testing and computing a certain chosen metric. Thus, at the end, each fold has been used for testing. All n results are then averaged, giving us the cross validated metric. If we consider n to be equal to the number of data points in our set, then the approach is also called leave-one-out cross validation. Thus, at each step we use all but one data point for training and one data point for testing. This method is however very expensive from a computational point of view, particularly in large datasets.

2.2.5.1 *Classifier Performance Metrics*

In what follows we look at the most common classifier evaluation metrics used in mobile sensing, as well as the selection process for the metrics we used for our own approaches.

An obvious choice as a metric would be accuracy, due to both its simplicity and its pervasiveness. It is used by a significant number of mobile sensing researchers [2,

34, 74, 75, 86, 87, 97, 123], being to the best of our knowledge the most used classifier performance metric in the field. Accuracy is defined, as shown by Equation (6) [140], in terms of true (TP) and false positives (FP), and true (TN) and false negatives (FN). Thus, for a multi-class problem like ours, one calculates the accuracy for all individual classes and then averages these values.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

Thus, for each class we determine the percentage of samples that were correctly classified as belonging to that class together with the samples that were correctly classified as belonging to other classes from the set of all classified samples.

Accuracy can however have offer a slightly limited understanding of results, since it considers true positives and true negatives together. If the dataset is unbalanced, an thus the number of negative samples significantly surpass the positive samples the results could be skewed, accuracy being high despite even all true positives being misclassified. More generally, as Provost et al express it [114], high accuracy could just be the result of either cost effectiveness, i.e., tuning probabilities of each class prediction, or of dataset class distribution manipulation. Typically on the x-axis of a confusion matrix is the predicted class while on the y-axis is the actual class. Thus, the true positives and true negatives are on the diagonal of the matrix, each of the elements of the diagonal constituting the true positive for its respective class, while the sum of the rest of the elements from the diagonal constitutes the true negatives. Therefore, accuracy can also be calculated as the sum of the elements from the diagonal of the matrix, divided by the total sum of the matrix elements, as shown by equation 7. As an example, if the dataset is unbalanced, most elements belonging to class A, one could use an algorithm that always predicts class A as result and still achieve good accuracy.

$$Accuracy = \frac{\sum_{i=0}^{n-1} M_{i,i}}{\sum_{i,j=0}^{n-1} M_{i,j}} \tag{7}$$

Therefore, given the potential issues of accuracy, detailed by [114], we have decided to use precision and recall instead. These two metrics could give us better insights about classifier performance for each class, instead of just knowing how many elements were correctly classified. The precision indicates what percentage of the total number of elements marked as class A actually belong to class A. Recall on the other hand indicates what percentage of elements belonging to class A were actually retrieved. Precision and recall are defined by equations (8) and (9) [140], while equations (10) and (11) define them for a specific class if we calculate them starting from the confusion matrix. While for our scenarios these two metrics are rather nice to have, there are scearios where their use is crucial. For a computer vision approach which aims to find tumors in medial images, recall is the most important metric. Thus, all tumors should be ideally identified, while a few false positives are not that harmful, since for all positives future investigations would be carried out. On the other hand, an approach predicting hotel ratings will aim for high precision: it is not

of much concern if a few good hotels will be overseen, but all hotels rated as good and therefore booked by customers should have the predicted quality.

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

$$Precision_i = \frac{M_{i,i}}{\sum_{j=0}^{n-1} M_{j,i}} \tag{10}$$

$$Recall_i = \frac{M_{i,i}}{\sum_{j=0}^{n-1} M_{i,j}} \tag{11}$$

Precision and recall are used as classification metrics by [38, 74, 75, 88, 90, 157] as well. However, for certain approaches we include accuracy values as well, in order to facilitate the comparison with certain related works, e.g. [87]. We occasionally replace precision and recall by a balanced F-score, which brings precision and recall into a single value. It is calculated as the harmonic mean of precision and recall, as shown by equation (12) [140].

$$F\text{-}score = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} \tag{12}$$

2.3 STATISTICAL METHODS

In what follows we explain the statistical methods we have used in our user study regarding disturbances caused by phones. In order to analyze the data, we leveraged three types of statistical operations in this study. Thus, we used the t-test to compare mean values. Since this test has to be performed pairwise, we first used the Analysis Of Variances (ANOVA) and Fisher's exact test to detect whether there are differences in means of more than two groups. We additionally looked into correlations to determine if there are any relations between two variables.

A key step before running any tests and making inferences is to establish the null and alternative hypotheses. The null hypothesis, denoted as H_0, generally corresponds to no difference between groups [118] or no correlation between phenomena, which can be denoted for instance by a variable mean being equal across the different groups. For this reason, in most cases one strives to reject the null hypothesis in order to prove the alternative hypothesis, or H_1. The alternative hypothesis is usually what a test aims to prove, such as being a difference between groups or correlation between phenomena.

It should be noted that the conclusion of the test is always expressed in relation to H_0, not H_1. Moreover, rejecting the null hypothesis does involve accepting the alternative hypothesis while not being able to reject the null hypothesis says nothing about the alternative hypothesis. The latter case could be caused by simply having insufficient evidence against H_0 and thus we cannot reject the alternative hypothesis based on it.

In the case of two groups, the null and alternative hypothesis would be usually expressed as follows [133]:

$$\begin{aligned} H_0 &: \mu_1 = \mu_2 \\ H_1 &: \mu_1 \neq \mu_2 \end{aligned} \tag{13}$$

The significance level of a test, denoted as α, represents the probability of wrongly rejecting H_0. This value should obviously be as little as possible, however its value is set by the researcher himself, usually based on existing conventions, be it general conventions or conventions pertaining to the specific test being used. A general convention in social sciences deems $\alpha = 0.05$ as sufficient. The probability value, or p-value, is "the probability of getting a value of the test statistic as extreme as or more extreme than that observed by chance alone, if the null hypothesis Ho, is true" [31]. Thus, the p-value is compared with the significance value, and if $p \leqslant \alpha$ then the result will be considered significant.

The two samples t-test is a hypothesis test to make assertions about "the mean where the data are collected from two random samples of independent observations, each from an underlying normal distribution" [31]. Thus, it can be used to determine whether there are significant differences between two group with regards to a certain variable. This test can only be used to compare two groups. For more than two groups, all combinations have to be compared pairwise. Therefore, it is more efficient to first use ANOVA, Kruskal-Wallis, or Fisher's Exact test to first determine whether there are any significant differences at all between any groups and only in that case to perform pairwise t-tests.

The ANOVA tests whether, out of three or more groups, there are at least two means of a given variable that are different from each other. It is based on the variances between and within the groups. If there are at least two means that differ, all means have to be compared pairwise to determine where the difference lies. This test requires a normal distribution of the values of each group, together with all variances being equal. There are however authors disputing the impact of data not following a Gaussian distribution on ANOVA test results, such as [82].

Fisher's test is particularly useful when checking for differences in two or more groups with regards to categoricalvariables. Additionally, this test has the advantage of being applicable even for small sample sizes. For large amounts of data, χ^2 could be also used, however since it just approximates instead of calculating the significance value α, it is not applicable for smaller sample sizes. Thus, the Fisher test checks the independence of variables from another variable, based on a contingency table. To do that it computes more extreme distributions with the same marginal distribution, then it calculates the probability of the given result together with the probabilities of the other more extreme results. The sum of all these probabilities gives the p-value. Similarly to ANOVA and Kruskal-Wallis, subsequent t-tests have to be carried out to find the differing groups.

The last type of operation we carried out was the determination of correlation between two variables. For this, we relied on Pearson's product-moment correlation coefficient, also known as Pearson's r. Pearson's r "is a measure of the linear dependence between two variables X and Y, giving a value between +1 and -1 inclusive" [49]. Pearson correlation for a variable can be computed as shows in Equation 14. There are three possible types of outcomes: positive correlation, negative correlation,

and no correlation. $r \in [-1, 1]$, with $r = 1$ corresponding to a perfectly positive correlation and $r = -1$ means a perfectly negative correlation. $r = 0$ means there is no correlation between the two variables. The interpretation of r is as follows: r^2 is the share of variance of one variable which can be explained through the other variable. Afterwards, a t-test must be used to check whether r is significantly different from zero.

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{14}$$

where $x_1, ..., x_n$ and $y_1, ..., y_n$ are the samples for variables x and y respectively, and $\overline{x}$ and $\overline{y}$ the means for variables x and y respectively.

3

RELATED WORK

In the current chapter we present the state-of-the-art in the field of smartphone sensing and context detection together with the open issues which served as starting point for the current work. We start with an overview of smartphone sensing and its applications. We consider the phone's position and the user's activity as the most relevant aspects for adapting the phone mode. Therefore, we subsequently narrow down our presentation to approaches aiming to determine phone position and then proceed to applications focusing on user activity. Afterwards, we provide an overview of existing applications that deal with disturbances caused by phones either by adapting phone mode or by controlling the routing and timing of the disturbance sources themselves. Finally, we highlight the open issues left and our approach to solve them, together with our main contribution.

3.1 SMARTPHONE SENSOR-BASED CONTEXT AWARENESS

Smartphone sensing has seen a great increase in popularity in the past years thanks to both availability and increase in phone performance. Previous approaches using wearable sensors or sensor platforms have already targeted a variety of application scenarios, but have met some significant problems: the overhead caused to the user by having to specifically carry extra devices, together with the extra costs incurred by these devices. Nowadays a significant part of the population in western countries owns a smartphone that they carry around, more specifically 64% of the population in the US [11] (in 2015) and 76% in Germany [8] (in 2016), making smartphones a suitable candidate for replacing wearables. However, there are issues still to be considered. One issue is the quality and variety of sensors incorporated in smartphones, together with the computational capacity of the smartphone itself. All these aspects have seen a significant improvement over the last years, making smartphone sensing suitable for a whole array of applications. An additional challenge is provided by the fact that wearable sensors can be placed in optimal places, such as around the knee when measuring aspects pertaining to running or walking, or on the user's collar when recording speech. Smartphones are not only unable to take advantage of optimal positioning of sensors, but are often disadvantaged in this regard. Users will carry their phones wherever it suits their needs and habits better, without regard to the fact that most accelerometer-based applications cannot function properly if the phone lies on the table, or that microphone-based application are severely hindered by carrying the phone in a backpack. This issue has two consequences: application scenarios might be somewhat limited and novel methods have to be developed to respond to the drop in accuracy. Some approaches even rely on using smartphone sensors together with one or a few additional external sensors in cases where accuracy is crucial. In what follows in the current section, we will provide an overview of application fields and scenarios for smartphone sensing without aiming to offer a

comprehensive view. We will cover the main directions in the field, show its breadth and its main applications.

3.1.1 *Application Areas*

3.1.1.1 *User Mobility Pattern Detection and Prediction*

One major application area concerns monitoring user movement, extracting mobility patterns and making predictions for the future, which is done mainly by relying on WiFi traces, occasionally corroborated with GPS, Bluetooth, or other sensor traces. Monitoring user paths outdoors is in most cases trivial, however for indoors there are a plethora of approaches to improve localization. First of all, one has to figure out if the user is indoors or outdoors. There are methods to solve this problem that rely on: 1) GPS and its reliability and (in)ability to get a lock [155]; 2) cell signal, light sensor, magnetometer data classification [163]; 3) all sensors in the phone, employing a semi-supervised classification mechanism [115]. Indoor localization has garnered particular interest which lead to a great variety of approaches, out of which we name just a few. [155, 161] use smartphone sensors alone such as the accelerometer, gyroscope, and/or light sensors, while [109] corroborate sensor data with automatically generated floor plans. Several approaches [20] resort to active probing by having smartphones or infrastructure devices that generate sounds for error correction and system calibration. [56] propose factoring in social encounters as means to correct errors and improve accuracy of indoor localization techniques. Azizyan et al. [2] aim to improve localization accuracy by taking into account a location's "ambience", which they determine based on WiFi, microphone, accelerometer, camera, and light sensor readings. Such an approach is particularly useful when the user is close to the wall separating two different locales. There is a significant body of work dedicated to extracting mobility patterns from GPS, WiFi and/or Bluetooth traces, as well as predicting the trajectory of the user, the next places he will visit and the duration of said visit [127, 152, 153]. Specialized applications focus on tracking pedestrians [59] or monitoring groups of people in dense urban spaces [134].

3.1.1.2 *Daily Life Applications*

There are several sensing applications that target the monitoring and improvement of the city and community life. Rana et al. propose a solution to estimate noise pollution levels and to create corresponding maps [117] while focus on estimating pollution levels. [95] use crowdsensing to predict parking lot availability and make recommendations to drivers. Optimizing the process of getting a taxi both from the drivers' and from the customers' perspective was targeted by [160] who monitor the number of available taxis at each taxi stand (in a given city) as well as the trajectories of the taxis on the move to predict the number of free taxis that will pass the stands. They correlate this with traffic conditions and average speed around the stand and present the information to the user in order to facilitate their decision whether to take a taxi or not. [50] monitor taxi routes and passenger occupancy in order to discover areas of service disequilibrium within cities. [162] provide a recommender for taxi drivers and customers alike regarding where to pick up the next taxi or customer respectively and the route to get there by extracting the passenger's mobility routes

and the driver's pick up habits from GPS traces. [45] identify the current transportation means of the user based on accelerometer readings, while [125] use the phone's barometer to identify whether the user is idle, walking, or using any transportation means. [57] propose an audio-based method to count the number of people in a crowd. Chon et al. [17] use GPS and WiFi traces, together with opportunistically collected pictures and sound clips from smartphones to link the locations visited by the user with their purpose (e.g., shop, restaurant etc.). Mun et al. [93] propose an approach to estimate the environmental impact of the user, while Englert et al. [32] estimate the user's energy consumption based on the sounds produced by various appliances.

3.1.1.3 *Social Applications*

Another class of application target the user's social relations and social network behavior. The CenceMe application [85, 86] determines the user's activity, mood, and certain environment parameters and shares them on social networks. [147] determine participants in a conversation, while [76] clusters conversation groups and builds discussion networks with regard to time. Additionally, [66] take into account the interdependencies between user's behaviors, as well as social norms and learned behaviors in order to improve user activity classification accuracy.

3.1.1.4 *Healthcare and Fitness Monitoring Applications*

Health monitoring can also be improved with the aid of smartphones. [67] propose a method for cough detection, while [48, 146] develop approaches to analyze lung-related sounds and detect respiratory-related health problems. Li et al. [70] propose a system to detect if a person is falling. Several approaches target snoring detection and sleep quality monitoring [15, 41]. A way to monitor the users tooth brushing habits is discussed in [60]. Determining the user's stress or boredom levels was targeted by [74, 111]. [156] assess the impact of college students' workload on their stress, mood, sleep, and physical activities.

A special subset of health-related applications concerns motivating users to exercise, as well as providing feedback to them. Gamification is often used to increase motivation and user enjoyment [16, 68, 101]. Dutz [30] uses smartphone sensor data in order to find the optimal moment to send the user a reminder to exercise. While fitness armband and sensors built in fitness devices are very popular, there is no shortage of smartphone-based applications either. Hao et al. [42] develop an approach to monitor the user's running rhythm, while Kranz et al. [61] rely on smartphone accelerometer data to create a virtual fitness coach. Thus, based on the data collected they record the activities carried out by the user, they assess his skills and provide feedback. Park et al. [101] propose a more general approach that has multiple alternative sensor data sources, alongside the smartphone sensors, and which transforms training intensity extracted from various types of exercising into game inputs. Several approaches [16, 68] monitor users while swimming and attempt to make their exercising more exciting by translating movements to game inputs such as actions in hunting a marine monster.

Having listed just a few application areas of smartphone sensing in order to provide an impression of the breadth of the field, in the following sections we shall focus

on determining specific contextual information such as the phone's position and the user's activity.

3.2 PHONE POSITION DETECTION

One of the earliest approaches aiming to determine phone position amongst other aspects belongs to Schmidt et al. [129]. They distinguish between five phone states, two referring to the phone position (hand and table), two to the phone state (silent and general) and one to the environment (outside). Being one of the earliest approaches, their goal was to prove the feasibility of determining context based on hardware sensor readings and of thus improving the user's interaction with the applications on his devices (phone or PDA). Their approach relies on a phone-connected sensor platform including two accelerometers, a photodiode, temperature and pressure sensors, a CO gas sensor and a passive IR sensor.

Later, Miluzzo et al. [87] propose an application to distinguish between two broad categories of phone position: inside (a bag, a pocket, etc.) and outside (in the hand, on a desk, etc.). Their envisioned proposed approach relies on the microphone, camera, accelerometer, and gyroscope incorporated in a smartphone. However, they implement and test their approach only using microphone-captured environment sounds. Their application achieves accuracies of up to 84%.

Several subsequent approaches use accelerometer readings to determine phone position [38, 135, 158], however rely only on passive probing of the environment. Fujinami et al. [38] classify nine phone positions based on accelerometer readings with a maximum accuracy of 74.6%. They consider three main phone positions: pocket, bag, and on a band around the user's neck. For these positions, they consider a few sub-positions by distinguishing between jacket and trousers pocket, or between backpack, shoulder and messenger bag. Wiese et al. [158] consider two sets of phone positions. One of them is similar to that of Miluzzo et al. [87], the authors distinguishing only whether the phone is inside or outside. The second class set sets apart three positions from the "inside" class: pocket, bag, and car phone container, while leaving the "outside" class undifferentiated (lumping hand, table, etc. together). We shall focus on the second approach, which is more interesting due to the higher granularity of phone positions considered. The authors propose two approaches, one that relies only on the smartphone's built-in accelerometer readings, which achieves accuracies up to 75.4%, and another approach that relies on several external sensors such as an external two-dimensional capacitive array and a multi-spectral sensor, which achieves a maximum accuracy of 85%.

Cho et al. [14] propose an approach relying on the vibration motor and accelerometer readings for determining the type of surface the phone is on. Surfaces include: sofa, plastic or wooden table, trouser pocket, hand, and backpack. It should be noted though that while some positions are determined with great level of detail (wooden or plastic table) some remain too specific (trouser pocket) missing a good number of similar but slightly different positions (any other type of pocket). Since they use active probing, the window length is an important issue which impacts the potential user disturbance. Authors use 1.3 seconds windows, which can cause a slight delay in the application response and use SVM as classifier, achieving an average accuracy of 85%. Shafer [135] proposed a method using also using the phone's vibration mo-

Work	Phone positions	Sensors used	Only phone sensors?	Maximum accuracy
[129]	hand, table	2 accelerometers, photodiode, temperature, pressure, CO gas, and passive IR sensor	×	-
[87]	inside, outside	microphone	✓	84%
[38]	pocket: chest, jacket, trousers (back & front); bag: backpack, handbag, messenger, & shoulder bag; around neck	accelerometer	✓	74.6%
[158]	pocket, car, bag, out	1) accelerometer 2) + light & proximity sensor external 2-dimensional capacitive array & a multi-spectral sensor	1) ✓ 2) ×	1) 75.4% 2) 85%
[14]	plastic & wooden table, trouser pocket, sofa, hand, backpack	accelerometer	✓	85%
[135]	table: kitchen, dinner placemat; desk home	accelerometer	✓	<70%

Table 1: Sensing modalities and phone position sets considered by related work

tor and accelerometer readings to classify phone positions such as table/desk, seat, or pocket, together with some extra aspects concerning the user's activity, such as walking or having dinner. However, he evaluates his approach only for distinguishing between four classes: desk home, table dining, table kitchen and table placemat with an accuracy of under 70% (exact number not reported, just a results plot). Thus, his approach ends up not really distinguishing between phone positions but rather user activities or environment aspects.

To sum up, there is a limited number of approaches targeting phone position. As shown by Table 1 these approaches either distinguish only a limited number of very general phone positions [87, 129], e.g. inside/outside, or despite the larger number of considered phone positions, they miss essential phone positions [14, 38, 158]. For instance, [38] classify different types of bags and pockets, but completely ignore the case when the phone is in the user's hand. Similarly, [158] ignores the user's hand. [14] at first glace consider all important phone positions, however some of them remain too specific and therefore not covering the entire class set, such as trouser pocket. Thus, in all the other cases when the user carries the phone in a pocket that

is not a trouser pocket, the phone position would be misclassified since there would be no correct class for it to be assigned to. Additionally, all existing applications either have accuracies of up to 85%, as shown by Table 1 which would still require improvement in order to be feasible for a real-world deployment. This is even more important if one aims to adapt phone mode based on this, given the potential for disturbance and user discomfort. For the record, best audio-based results are achieved by [86] with 84%, while best accelerometer readings based results are achieved by [14] with 85%. Furthermore, to the best of our knowledge, none of the existing approaches determines the four essential phone positions that we determined in our user study, pocket, bag, hand, and desk, as we show in Chapter 5.

3.3 USER ACTIVITY DETECTION

In what follows, we will focus on approaches aiming to determine user activity. Despite the great advances in smartphone sensing, this area is still insufficiently developed. It is often difficult to determine all relevant activities in a given scenario, since certain activities lack specific fingerprints that can be captured using sensors or have overlapping fingerprints with other activities. In this case, it is much easier to capture patterns of even complex activities both due to the convenient placement of sensors and the availability of a wider array of sensors. Let us assume we want to monitor a person's eating and drinking habits to make sure they eat at a healthy pace, get a sufficient caloric intake and drink enough liquids to avoid dehydration. Just determining these activities would be possible also using the smartphone's microphone. However, it would not be able to tell how much they've been eating or drinking, or whether that is sufficient. This could be realized using body worn accelerometers, placed, say, on their arms, which would offer us more information about the exact movements, together with physiological sensors which would provide insight into whether the patient is getting too many or too few calories, or whether he is getting dehydrated (e.g., based on the heart-rate). Let us additionally assume that we want to monitor the drink intake of a person that is a recovering alcoholic: Smartphone sensors could not reliably tell us the user picked up a glass, nor give us hints about the contents of the glass, whereas wearables could provide an accurate recognition. On top of this, for certain activities users often put their phones on a surface or leave them behind, e.g., when working at their desk or cleaning up their flat. Sometimes, additional methods could be employed to mitigate these disadvantages of smartphone sensing, however at the cost of privacy loss and thus low user acceptance. For instance, if one wants to determine learning-related activities, knowing that the user is typing is not enough. A smartphone-based option to solve this issue would be to recognize based on sound what letters are being typed and use natural language processing techniques to figure out what the topic of the text so they can draw conclusions about the purpose of the activity. For the above-mentioned reasons, we discuss in what follows both approaches that rely on wearable sensors as well as solely smartphone-based methods.

3.3.1 *Body-worn Sensors*

In what follows, we will focus on approaches that rely on wearable sensors or sensor platforms for user activity detection. We do not consider infrastructure sensors nor approaches relying on RFID tags attached to objects such as [100, 103, 108, 145] since they rely on the assumption that all spaces where the user performs certain activities will have certain sensors. Our work focuses on pervasive approaches that are user-centric and thus do not depend the on environment setup and are not limited to certain spaces.

A part of the approaches concerning user activity classification rely on a single sensing modality, or multiple sensors of the same kind. Chen et al. [12] propose an audio-based solution to monitor bathroom activities in order to understand personal hygiene habits of patients suffering from certain diseases such as dementia and to allow clinical practitioners to help them. Stager at al. [142] detect activities that can happen in a workshop such as hammering, sawing, grinding, or filing, as well as activities correlated to taking a break, such as grinding coffee and using the microwave. They rely on two microphones, one worn next to the user's chest and the other at his wrist to distinguish sounds that happen next to the user's hands from those occurring in the user's surroundings. Hung et al. [51] use a body-worn accelerometer to determine social interactions-related activities, such as talking, laughing or drinking.

Bao et al. [5] propose the first approach using between two (worn on the user's thigh and wrist) and five multiple external accelerometers to determine a set of physical activities of the user such as sitting, standing, walking, or running. To these, they add various daily activities that have a distinct motion fingerprint, such as folding laundry, eating/drinking, or brushing teeth. Choudhury et al. [18] introduce the Mobile Sensing Platform (MSP), a platform which can be worn as a belt clip and which consists of eight sensors: an accelerometer, a compass, a microphone, a barometer, as well as humidity, visible and infrared light, and temperature sensors. MSP obtained superior recognition accuracies for the "usual" physical activities considered for instance by [5], as well as for selected daily activities, such as working on a computer, eating or talking. [80] built a watch containing an accelerometer, microphone, as well as a light and temperature sensor. Using it they explored the impact of sensor positioning (e.g., wrist, ankle) on physical activity classification performance. [102] propose an automated physical activity diary which provides the user summaries and feedback and allows him to reflect on his exercising patterns. In the same direction of encouraging users to exercise more, [19] follows the user's patterns over time using MSP and uses gamification aspects to get them to increase the time spent doing sports or the intensity of the workouts.

Ward et al. [157] rely on the continuous sampling of body worn microphones and accelerometers to determine any activities that would have a specific movement or audio fingerprint. They propose as target for their application assembly and maintenance workers. In their prototype, they consider the daily activities in a "wood workshop", i.e. hammering, drilling, sawing, opening a drawer etc.

Mesaros et al. [84] rely on audio recordings as well to recognize various sounds which correspond to activities such as riding the bus, being in a shop or at a baseball match. They do not consider a complete activity set for a certain scenario, but rather classify as many different sounds as possible in a greedy fashion.

Unlike the previous approach, [71] build a hierarchical activity model, splitting all activities into two main groups: navigation activities and significant activities. They collect just GPS traces and infer based on that the user's significant places and then model the sequences and correlations of the places and activities of the user. Based on this, they classify the user's activity and make predictions about its duration and the activity that will follow up.

All afore-mentioned approaches rely on supervised learning. There are attempts to determine daily activities on a larger scale using unsupervised learning. In order to still achieve a satisfactory accuracy, authors either detect only low-level activities [89] or employ supervised learning to determine low-level activities and then discover higher level activities in an unsupervised fashion [52]. It should be noted though that the activities they deem as low-level, e.g., brushing teeth, eating at the table, or hovering, is in disagreement with a number of authors in the field [18, 84].

As previously discussed, these approaches have the advantage of collecting less noisy data than sensors incorporated in smartphones and capturing more relevant features (like collecting data from multiple microphones placed in strategic locations). However, they also require external hardware, which incurs extra costs and can cause user discomfort and heightened privacy concerns. All other drawbacks aside, these approaches require that the users carry extra devices on a daily basis, which hinders user adoption and acceptance. For these reason, in what follows we will discuss approaches that rely on a device that most users carry around the whole day: their smartphone.

3.3.2 *Smartphone Sensors*

In what follows we will group the approaches, as we also did in the previous sections, by application scenario. An alternative would be to group them by the sensors used, especially since there is a limited set of available sensors in smartphones. However, sensor selection is in most cases a consequence of the class set to be determined, which is in turn is determined by the application scenario. There are however a few applications which consider a certain sensor and aim to distinguish as many different user activities as possible. We present these applications at the end of the current section.

A significant amount of research was put into physical activity classification and its applications. The initial sensors used were the accelerometers and gyroscopes incorporated in most smartphones, but as application scenarios branched out, more and more sensors were included. First approaches aimed to recognize a set of common user movements, such as walking, running, going up- and downstairs, sitting, and standing [63, 94, 159]. It might appear surprising to label sitting or standing or movements, but even in these situation the user presents certain patterns of movement (e.g., moving from one leg to the other or making small steps when standing), which allow the classification of these activities. Some authors, like Keally et al. [58] extend the physical activity set to include not just movements, but also other daily activities which have characteristic motion fingerprints, such as cleaning, eating, or watching TV.

There is a great range of applications making use of the automatically determined physical activities. One class of applications includes sports and fitness applications.

[42] develop an approach to monitor the user's running rhythm while Kranz et al. [61] propose a prototype for a personal fitness trainer. Thus, their application assesses the user's movements and based on this provides feedback and recommendations to the user. Additionally, they attempt to motivate the user to exercise more. Mitchell et al. [90] classify the main movements present in games of soccer and hockey in order to allow a subsequent evaluation of the performance of the players. Lee et al. [68] propose a dragon-fighting game which receives as inputs the different types of strokes inferred based on smartphone accelerometer (connected by a band to the lower back), which get different meaning in the game. Choi at el [16] propose a collaborative exergame that brings swimmers together to hunt a virtual monster. They rely on the accelerometer, gyroscope, and barometer data from the waterproof smartphones which users carry on armbands to determine their stroke and speed.

Worth noting are also medical applications such as determining whether a person is falling [70] or assisting Parkinson patients in keeping their posture [81]. Another category of medical applications rely mainly on the smartphone's microphone to determine whether the user is coughing [67] or assessing the user's sleep [15, 41]. The latter type of application concerns itself with sleep "activities" such as snoring or breathing normally. Additional attention is given to activities relevant for a good health such as proper tooth brushing [60].

The CenceMe application [85, 86] determines user activity, together with the user's mood and certain parameters regarding the state of the environment (e.g., noisy, hot, bright). They differentiate between simple activities (e.g., sitting, meeting friends) and habitual activities (e.g., at the gym, at work). Their goal is to share these elements on social networks, as well as on instant messaging applications. For their approach, they record and classify GPS, accelerometer, camera, and audio data. They rely on smartphones as well as embedded sensor platforms and recreational sensor platforms for collecting the sensor readings.

There is a whole group of approaches that do not focus specifically on user activity determination, but rather on capturing certain aspects from the surrounding environment. However, since some of these aspects are directly correlated with the user's activity, we offer an overview of these papers in what follows. Rossi et al. [123] rely on audio recordings to classify environment sounds such as those caused by a coffee machine or a shaver, or by being in a restaurant. They limit themselves to contexts which have a distinct acoustic fingerprint, without aiming to have a complete class set for a given problem. Hemminki et al. use accelerometer traces to determine the transportation means that the user is currently in [45]. In [32] authors detect the running electric appliances situated in the proximity of the user based on acoustic fingerprints. Their goal is to estimate the users' energy consumption, however running devices such as the TV, over, or water cooker can give strong indications regarding the user's activity. The SoundSense application [75], besides classifying the genre of the music played in the proximity of the user, also detects user activities which imply speaking, such as chatting, being in a meeting, or reading something out loud. Additionally, they identify several environment sounds which are connected to user activities, such as driving, being on a plane, or vacuuming. Once more, the focus lies on correctly classifying as many acoustic fingerprints as possible and not on building an activity model and then detecting all corresponding elements.

Lane et al [66] enhance activity classification by factoring in the social influences by making two observations. First of all, connected people will have interconnected

schedules, like common meetings. Secondly, each community has its behavior norms to which its members abide. Thus, they build a hierarchical model in which they consider the communities the user is a part and build a multi-layer model consisting of the community and user behavior layers. They perform the activity classifications separately on each of the layer, then use an inference process to decide on the actual activity.

While the afore-mentioned approaches rely on supervised learning, there have also been attempts at unsupervised or semi-supervised approaches. Finding users and collecting sufficient high-quality data from training the activity classifiers are a real challenge which these applications try to address. However these approaches present a marked drop in accuracy compared to similar supervised approaches, e.g., the unsupervised part of the SoundSense application [75]. Attempts to mitigate this include semi-supervised approaches [77], however only with limited success and still significantly lower accuracies compared to supervised training approaches. An additional issue is that data is often mislabeled due to lack of attention or to differences in understanding a situation. For instance, users might label an activity as "dinner" both when eating at home and in a restaurant, despite the obvious difference. Approaches such as that of Peebles et al. [105] try to solve this labeling problem by using similarity metrics and clustering labeled activity data in order to correct mislabeled samples and distinguish between ambiguously labeled samples. They apply their method to four main categories of activity: work, drive, transport and shopping. Each of these activities hash additional parameters to be classified, such as working, office, and name of the building for the activity "work". [64] implement feature sharing between users in the same social circles, which is a precursor of [88], missing its classifier adaptation capability (applied by the latter to a different scenario and not activity classification though).

To sum up, there are a plethora of approaches to determine user activity, however most of them rely on wearable sensors due to the ability to place sensors in strategic positions in order to enhance accuracy, as well as the endless available choice in sensors. Smartphone-based approaches are limited by the positioning of the phone and limited array of sensors, so they often focus on determining all activities they can, without obtaining a complete class set for a certain scenario. However, determination of all relevant activities for a relevant scenario remains an essential goal for any real-life application. Therefore, the remaining challenges for our thesis will be to decide on the relevant class sets for our scenarios and then apply and adapt the corresponding methods from the related work.

3.4 APPROACHES TO AVOID THE DISTURBANCES CAUSED BY SMARTPHONES

The multitude of applications enabling communications between users has created, besides the obvious benefits, a series of problems. First of all, it lead to soaring number of daily interruptions, which have a significant cost on both the completion time for the tasks during which interruptions come [1, 3, 22, 47, 53] and on the user's emotional state [4, 164]. The around-the-clock availability brought by smartphones, brought additional challenges to preserving a work-life balance. Last but not least, incoming interruptions are often signaled acoustically, be it the ringing for phonecalls or the email client beep sounds for incoming emails.

Ho et al. [47] carry out a survey of the related work, based on which they highlight eleven of the most important factors that determine the impact of the interruption. These factors can be further grouped in three categories: 1) aspects concerning the user, such as his current activity and his engagement in it, together with the previous and the future activities, the user's emotional state, his previous pattern of reacting to disturbances; 2) aspects concerning the message, such as its utility and the time it takes to understand and respond to it; 3) aspects concerning the disruptions management, such as the way they are delivered and the perceived control over how they are delivered, as well as the frequency of disturbances. The first and the third class of factors can be tackled through the design of the notification delivery mechanism, whereas the second class of factors cannot be controlled and can just be taken into account in the timing of notifications to minimize their impact.

There are two main ways to approach the issue of minimizing the impact of interruptions. The first category of applications attempt to adapt the device's notification mechanisms. Thus, depending on the user's context, he might be notified in a more subtle way or might not be notified at all about an incoming source of disruption. The other way to approach this problem is to schedule and control the way incoming phonecalls/notifications reach the phone. For instance, messages might be routed to different users in the system or withheld until the user switches to a situation which allows disruptions. This category of applications can also take into account the importance of the incoming interruption source, not just the interruptibility degree of the user's current activity. The majority of existing approaches fall into the second category. We should note that in the related studies the words "availability" and "interruptibility" are often used interchangeably, which we shall do as well in what follows.

3.4.1 *Adaptation of the Notification Mechanism*

Amongst approaches that deal with daily interruptions from communication applications, those that adapt notification mechnisms, for instance by setting the volume of the notifications or muting them, are in a clear minority. This general idea is also used in broader platforms [62, 131], which centrally manage multiple disturbance sources such as email, chat and telecommunications applications, phone etc.

Tsai et al. [148] target the potential discomfort caused to the user when he picks up a phone which is in the normal or loud mode. Thus, they avoid the user being disturbed by the loud ringtone until they pick up the call by gradually reducing the ringtone volume from the moment they touch the phone to that of answering the call. Their solution's accuracy is strongly influenced by the position of the phone.

Siewiorek et al. [138] propose an approach to determine the user's availability and based on it adapt the ringer volume and usage of vibration motor, i.e. phone mode. They rely on a number of wearable sensors such as voice and environment microphones, accelerometers, light and temperature sensors to classify four states of the user: uninterruptible, idle, active, and normal (the default option). In addition to adapting the phone mode, their application sends an automatic SMS to all callers if the user is uninterruptible, offering them an option to still get their phonecall to the attention of the user if the issue is urgent.

3.4.2 *Prediction of the User's Willingness to Receive Phonecalls or Messages*

We shall present in what follows an overview of approaches that handle interruptions by timing the delivery of the interruption source to match the user's availability. Thus, despite following a different approach to dealing with interruptions, they still follow the same goal, namely minimizing user disturbance and making sure interruptions are only received when the user available.

There are two essential aspects in timing the delivery of interruptions: establishing that the user is not interruptible at the moment and determining the optimal moment to deliver notifications. Several studies investigated the optimal moments to deliver an interruption. Bailey et al. [3] show that interruptions during a certain task lead to a sensibly longer time to complete the task and that transition moments between activities seem to be an optimal moment to deliver disruptions. Several studies [1, 47, 53] show that delaying interruptions until the user reached a breakpoint in his tasks can significantly mitigate the effects of interruptions, reducing both the user's response time and frustration level [55]. There are three types of breakpoints [53]: fine, medium and coarse, and interruption effects are minimized if the notification is deferred until a coarse breakpoint is reached [53]. Coarse breakpoints can be mapped to the activity transitions defined by [3]. Harr et al. [44] argue for the role of social situation in the user's decision and attitude towards the incoming interruption.

Thus, a whole class of approaches delivers interruptions during transitions between activities. Building upon the findings of [3], Ho et al. [47] propose an approach for PDA devices where message delivery is timed to match the user's transition between activities. Activity transition is detected based on posture changes which are captured by several body-worn accelerometers. Based on the findings of [1, 47, 53], Iqbal et al. [54] propose a statistical approach to determine all three types of breakpoints in PC desktop environments. However, most approaches [35, 37, 83, 106, 110, 113, 126] still consider all moments, including in-task times when calculating the user's interruptibility assuming certain activities to be less affected by interruptions or simply less important. This presents an obvious advantage in case the user receives too many notifications which would make their delivery between tasks difficult, since too many notification would pile up for each transition.

Aside from the approaches aiming to determine the optimal timing of the interruptions with regards to task completion time (e.g., by delivering interruptions in the transition between tasks) and user sentiment, there are a series of works that only focus on maximizing response probability and time, while optionally considering the user's mood. Thus, they consider the user's response as a validation mechanism regarding the moment when the message was delivered. Pielot [110] defines availability as the user picking up an incoming phonecall. He tries to predict whether the user will answer his phone based on mostly soft sensor data, such as last call and last screen change, the number of previous phone conversations with the caller, ringer mode etc. To these he adds physical activities determined based on accelerometer readings. Pejovic et al. [106] define availability based on three factors: whether a user will respond to a phonecall/message, the time to do so, and the sentiment towards the received interruption. They use machine learning techniques to infer these factors from user activity, emotions and engagement, together with the location and time of the day, which in turn they obtain either from the smartphone sensors or through users' self reporting.

Another class of applications leverages machine learning techniques to determine the most relevant features and then classify the user's availability. Sarker et al. [126] focus on determining the features with the highest correlation to user availability and train classifiers using those features. Thus, instead of using response time as indicator of availability, they collect the users' self reported availability status, together with an extensive list of features. They then determine the features with the highest correlation, narrowing down the list from 99 to 30 features, including the user's activity and stress level, as well as the time and day of the week. Data was gathered from smartphone sensors, wearable sensor platforms, as well as through self reporting, e.g. for stress levels or availability at the given time. The authors' goal is to support abstinent smokers and prevent them from lapsing by delivering messages in critical moments, while still taking into account the user's interruptibility in order to maximize the success chances of the intervention. Mehrotra et al. [83] consider a series of additional features regarding the application that issued the notification, the relationship between the sender and the receiver, as well as the content of the message for classifying the user's interruptibility.

The afore-mentioned approaches manage incoming interruptions around the clock. Some approaches however focus only on situation when this is more likely to be needed, for instance in the office [37] and therefore rely on infrastructure sensors as well in order to determine the user's activity and availability. Turner et al. [149] provide a survey of existing works regarding timing of interruptions, looking at the different ways to define interruptibility, gather and label data, incentivize users, as well as select relevant features and classifiers.

In the following section we sum up our findings regarding existing approaches dealing with disturbances and highlight the open challenges, focusing on the degree of automation and user involvement and overhead in determining the interruptibility degree. We additionally highlight which of the approaches and to what extent deal with and are fit specifically for phone-generated disturbances.

3.5 OPEN CHALLENGES

One essential aspect for all papers dealing with disturbances is the determination of the various factors based on which availability is inferred. Thus, if the proposed solution concerns phone mode adaptation, the core issue is the detection of contextual information based on which to infer whether a loud, normal, silent, or no notification at all is appropriate. For solutions that time notifications, this issue refers to detecting context information that can point towards the appropriateness of a disturbance, e.g. by determining if the user is engaged in a certain activity or in transition between activities. There are four main data sources: 1) smartphone sensors; 2) wearable or infrastructure sensors; 3) soft sensors; 4) users' self reporting. While the first two data sources are used for similar applications, wearables are the most popular choice [47, 126, 138]. Thus, phones are mainly used to determine physical activities. Even transitions between activities are determined either by using software sensors [54, 55] or by using wearable accelerometers instead of the smartphone ones [47]. In case of using smartphone sensors, approaches are however limited by the fact that the user needs to be working on the same device that manages his notifications, otherwise if he carries out any other type of activity or receives notifications

Work	Data sources	Targets phone interruptions	Contextual information
[47]	wearable sensors	×	breakpoints in a task
[37]	infrastructure sensors	×	physical activity, state of environment objects
[54, 55]	software sensors	×	breakpoints in a task, exemplified with PC activities
[35]	phone sensors	✓	GPS location, noise level, presence of voices, time
[138]	wearable sensors, software sensors	✓	calendar, physical activity, noise level, user's voice detected
[110]	software sensors	✓	caller and running application info
[126]	wearable sensors, self reporting	×	
[113]	self reporting	✓	physiological data
[106]	phone sensors, self reporting	✓	GPS location, physical activity
[83]	software sensors, phone sensors	✓	WiFi access point, noise level, physical activity

Table 2: Related work targeting interruptions: data sources and contextual information

through other devices disturbing interruptions cannot be suppressed. In the second case, there is the limitation that the user needs to wear accelerometers in strategic body positions in order to ensure that changes in posture and other subtle movement cues are accurately determined. Software sensor-based approaches [54, 55, 110] rely on running processes, previous user interaction with applications, device settings etc. User's mental state is either determined through self reporting [126] or by using wearable physiological sensors [126].

As Table 2 shows, most approaches do not deal with smartphone-induced interruptions. Additionally, solutions that time the delivery of notifications cannot be used for managing incoming phonecalls but only asynchronous forms of communication. We shall look in what follows at approaches that apply to smartphone interruptions, with the aim of avoiding both disturbances and missed phonecalls or notifications. Pielot [110] considers just software sensors which provide information just regarding the user's interaction with the phone and thus have limited ability to predict the user's availability. Poppinga et al. [113] rely exclusively on self reporting from users. Fisher et al. [35] consider just GPS location, whether any voice can be recognized in the proximity of the phone, together with environment noise level and time. [106] rely on hardware sensors and self reporting of users, however the contribution of sensors is minimal, referrring just to GPS-acquired location and physical activities. The more relevant aspects, such as actual user activity or type of activity (leisure, work, etc.) were manually filled in by the users. Mehrotra et al. [83] combine software sen-

sors with hardware sensors. However, they infer only low-level context information, which has little direct correlation with availability, e.g. whether the user is connected to the WiFi, whether the environment is noisy, and just the physical activity of the user, instead of a higher level activity. Additionally, they determine the user's proximity to the phone using the proximity sensors, which is an inaccurate solution since the proximity sensor will return a positive result given the proximity with any object, without any connection to the user. Sieworek et al. [138] rely exclusively on a wearable sensors platform, which implies a clear overhead for the users. The work of [47] concerns PDAs, but same methods could be applied also to phones; however, they also rely on wearable sensors for their approach.

To sum up, existing approaches that mitigate effects of daily disturbance deal with it by either timing the interruption delivery or adapting the notification mechanism so that the interruption would not be noticed at an improper moment. There are various factors which can be considered as indicators of interruptibility, however the contribution of context information inferred based on smartphone sensors is up to now minimal. Thus, existing approaches limit themselves to low-level types of context such as GPS location, physical activity or noise level of the environment. However, higher-level types of context such as phone position and user activity can provide more insight into the user's availability and remain up to now an unexplored opportunity in this field. For instance, the WiFi to which the user is connected can reveal that he is at work, but cannot tell if he is holding the phone in his hand, so a loud ringtone would be disturbing, or in his bag, where he would not hear notifications if the phone is on silent mode. Additionally, the user's activity can influence the appropriate phone mode: while working in his office, he might be willing to hear incoming notifications, while in a meeting one would typically set the phone mode to silent or "only lights".

3.6 OUR WORK

The goal of this thesis is to provide a solution of automatic smartphone mode adaptation so that the user can avoid both disturbances and missed phonecalls. In order to achieve this, we aim to classify the phone's position and the user's activity based on smartphone hardware sensors. We limit ourselves to the sensors incorporated in the smartphone and among those just to hardware sensors in order to make our approach accessible to as many users as possible and to minimize the amount of effort required of the user (e.g., by attaching and carrying additional sensors or by entering appointments in a calendar app). Additionally, we collect the sensor readings in an opportunistic fashion when a phonecall or notification comes in, in order to ensure that the classified contextual information is not deprecated and that the phone mode is adapted precisely when needed. The classification results are then used to adapt the phone mode using a predefined mapping, e.g., select "only lights" mode if the phone is on a desk during a meeting, which each user can afterwards personalize.

Our work on phone position detection is presented in Chapter 5. Different aspects of the implementation of phone mode adaption as well as our contributions with regards to user activity detection are presented in Chapter 6.

4

USER STUDY

4.1 GOAL OF THE STUDY

In the following chapter we present the user study we carried out to investigate the frequency of users' phone mode adaptation, together with the incidence of undesired effects of failing to adapt phone mode such as disturbances as missed phonecalls. Additionally, we aim to understand the factors that determine the users to change and decide the suitable phone mode. Last but not least, we intend to gauge the users' attitude towards phone mode adaptation solutions.

In what follows, we present the main findings of our study. We start with a general description of the survey and of the data gathering process. We subsequently proceed to answer the essential questions behind this survey: 1) What is the relevance of phone mode adaptation, i.e., how often people adapt phone mode, how often they fail to do so, and what are the consequences; 2) What are the factors that determine the appropriate phone mode and are there common preferences amongst users regarding the importance of those factors and the desired phone mode; 3) What are the users' attitudes towards phone mode adaptation solutions and, in case of a positive attitude, which of the different options for solutions – recommendations vs. automatic adaptation – are favored. Additionally, we include interesting correlations that were found and did not fit in previous sections and discuss their potential reasons and implications for our approach. The final subsection sums up our main findings and their implications.

We have excluded from this chapter the following aspects which will be discussed in the respective chapters where they belong thematically: the statistical methods we employed are described in Chapter 2, while users' preferred phone positions are included in Chapter 5.

4.1.1 *Survey Description*

The survey consisted of five parts including twelve short batteries of closed questions with mostly one open category for further comments and additions. The first part gathered demographic information about the users such as their age, gender, and occupation. The second part aimed to determine the phone settings they currently adapt manually. The proposed options were not limited to phone mode, including in addition screen brightness, phone mode, and presence information on communication apps such as Skype or WhatsApp. The participants had the option to input other options they currently adapt.

Section 3 concerns itself with understanding the factors that trigger phone mode adaptation, i.e. what determines users to change their phone mode, and what are the users' preferred phone modes based on the afore-mentioned factors. The following section investigates the frequency of user's manual phone mode adaptation, together

with the frequency of their and the surrounding people's failure to do so which resulted in disturbances and missed phonecalls. The last section refers to the users' openness towards approaches for automatic phone mode adaptation. The complete questionnaire can be found in Appendix A.1.

4.2 DATA COLLECTION AND CLEANING

The survey was carried out between May 25th and August 15th, 2016. It was an online study where the participants were found through the snowball principle, i.e. the link was shared with an initial group of people which were asked to further distribute the link to people they know. The initial group included project partners, colleagues and research associates from different labs, students from different universities (announced via the university forum), as well as friends. The condition to have representative results is to randomly and with the same probability assign every member of the population of smartphone users to the study. An additional concern is the margin of error of the results, which influences the required number of respondents. For instance, one would require about 1000 respondents to achieve a margin of error of 3% at 95% confidence. Since the effort to get a fully randomized sample was too high, this presented itself as the best option to gather sufficient data to make expedient statements about the mode adaptation of smart phone users. Additionally, this was meant to be an exploratory study that should shed a light on the behavior of smartphone users and it did not aim on representative results in the first place, so the snowball principle was a suitable decision.

The sample size was 96 participants. Out of these, six had to be excluded from the analysis since they either filled in the just demographics section of the survey or, as in the case of one specific respondent, they claimed to have no phone so the questions did not apply to them (additionally, they answered all questions with the very same rating – the lowest possible). Out of the remaining 90 respondents, 69 have completed the questionnaire. From those who did not finish the survey, 16 respondents answered the first four questions (which include only one non-demographic question), one answered eight questions, one ten, and another three respondents answered eleven questions. We included the answers of the respondents who did not finish the survey in the analysis of the respective questions that they answered. As stated in the beginning, the only respondents we excluded completely were those that completed only the demographics section. Among the interviewees there were 32 women and 58 men. The age of the respondents ranges from 18 to 72 years with a mean of 30.61 and median of 28 years. As far as the current occupation of the respondents is concerned, the distribution is as follows: 35 students, 31 researchers, 16 office workers, two stay at home parents, five unemployed, one manual worker. Given the low number of respondents falling in the last three categories, we decided to consider only the the first three occupations, i.e., students, researchers, and office workers for the analyses concerning occupation.

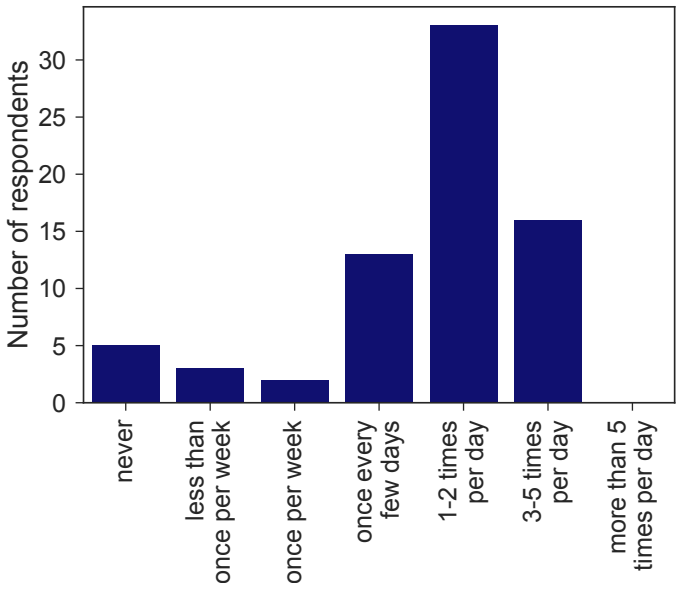


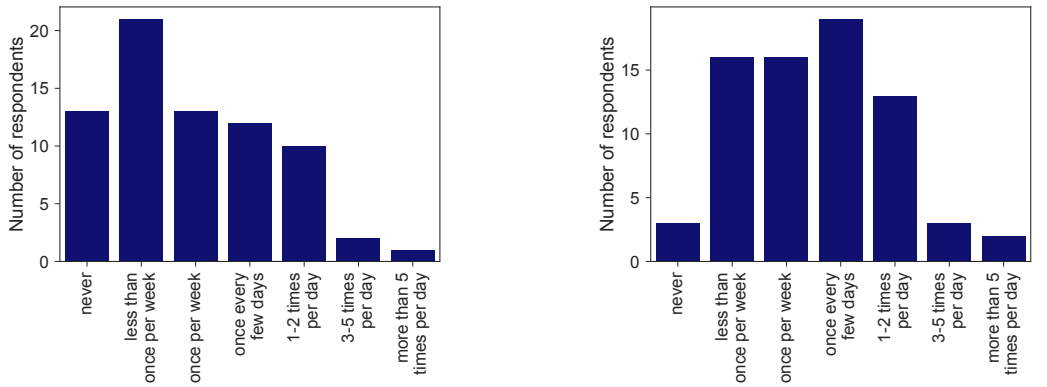
Figure 9: Frequency of manual phone mode adaptation done by the users

4.3 RELEVANCE OF AUTOMATIC PHONE MODE ADAPTATION

In order to assess the need for an automated phone mode adaptation solution, the first aspect we investigated was how often users adapt their phone modes throughout the day.

Figure 9 shows the distribution of frequency of phone mode adaptation amongst users. One can note that more than 70% of the respondents adapt their phone at least one or two times per day. Out of these, more than 30% adjust their phone mode three to five times per day (and 23% of the total respondents). At the same time, 18% of the users change their phone mode once every few days, while only 14% do it once per week or less often. Based on these findings, one can conclude that automated phone mode adaptation would be of benefit for a significant percentage of respondents. Furthermore, the free-input fields concerning the participants' attitude towards automated solutions revealed one reason why some users very rarely adapt phone mode: they keep their phone most, if not the whole time, on silent mode. While this is a very effective method to avoid disturbances, it also makes overlooking incoming phonecalls or messages much more likely, unless the user is particularly careful about the place where he carries and stores his phone.

Figure 10 presents side by side the frequency of phone mode-related disturbances caused by the user and by the other users respectively. The former case includes also the situations when the respective user was the only one disturbed by the interruption. A quick glance reveals that users report slightly more disturbances caused by other people than by themselves. The reasons behind that may include a tendency of users to be more lenient and underestimate their own mistakes but also the fact that users which pay closer attention to adapting their phone mode would be more likely to have an interest in filling out our questionnaire. Let us focus first on disturbances caused by the respondents. 18% of the participants create or just undergo a disturbance once a week, while 17% do this every few days. It should be noted



(a) Disturbances caused by the user

(b) Disturbances caused by other users

Figure 10: Distribution of frequency of disturbances due to the wrong phone mode

that 14% of the respondents cause a disturbance once or twice per day, while one of the respondents admitted to doing this on average five times per day. Even if we consider only users who are being disturbed at the very least every few days as our target, they still represent 36% of the total users.

More than a quarter of the users are disturbed at least once or twice per day by other users' phones, while 7% of them report this happening three or more times per day. 27% are disturbed every few days and 23% once a week. It should be noted that only 4% of the respondents are never disturbed by other persons' phones, which would make phone mode adaptation of little interest for them.

Therefore, the average number of times the user has to adapt phone mode every day argues for the development of an automated solution. Furthermore, disturbances

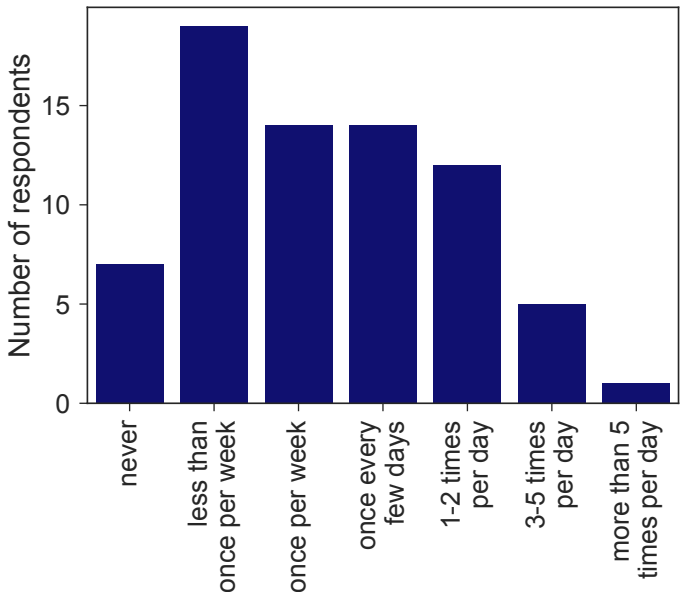
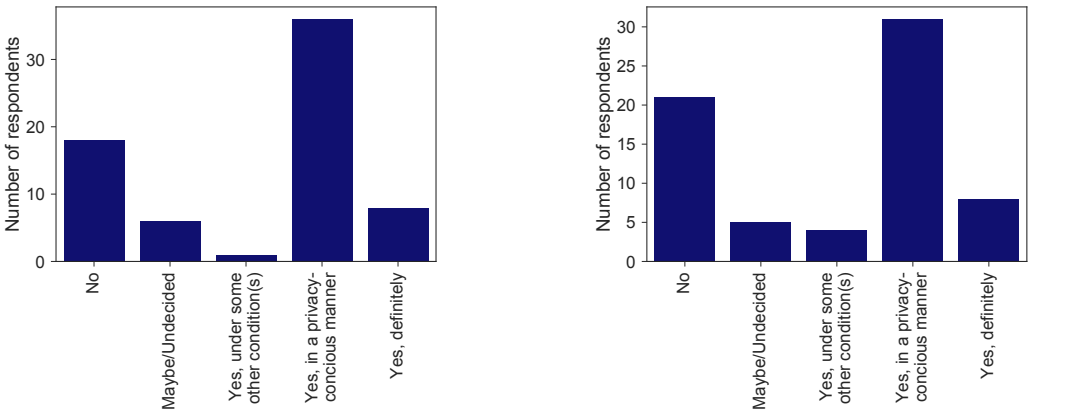


Figure 11: Distribution of frequency of missed phonecalls as a result of the wrong phone mode



(a) User acceptance of automated solutions

(b) User acceptance of adaptation suggestion

Figure 12: User acceptance of automated phone mode adaptation solutions vs. recommendations for phone mode adaptation

caused by phones ringing at the wrong times seems to be a pervasive problem. Conversely, leaving the phone most of the time on silent mode is not a solution either, leading to missed phonecalls. Figure 11 shows the distribution of missed phone calls due to the phone being in the wrong mode. All these aspects argue for the usefulness and need for an approach that relieves users from the need to adapt phone mode while allowing them to avoid missed disturbance and phonecalls at the same time.

4.4 ATTITUDES TOWARDS AUTOMATIC PHONE MODE ADAPTATION

We proposed two options to our respondents. In the first case, the application would automatically carry out the phone mode adaptation, while in the second case the user would simply get notifications when the phone mode is not suitable for the current activity.

Figures 12a and 12b present the users' attitude to the two approaches to phone mode adaptation. One can notice a slight preference towards the automated solutions. Thus, more than 10% of the respondents would be willing to use such an approach without having any additional concerns, while in total more than 60% of the participants would be willing to use an automatic phone mode adaptation approach under certain conditions such as privacy awareness. A quarter of the remaining users are undecided and the remaining three quarters (26% of the total) would not use such an approach. At the same time, slightly less users would accept an application that would provide suggestions regarding phone modes, and slightly more users would completely dismiss such a system compared to the automated solution.

4.5 OFFICE WORKERS' ACTIVITIES AND CORRELATIONS TO THE APPROPRIATE PHONE MODE

We additionally investigated whether there are any general user preferences regarding the mapping between certain activities and the corresponding phone modes. As previously mentioned, we proposed a set of activities that we deemed as most com-

mon for office workers and most likely to require a phone mode change, but respondents had the choice to add their own activities to our list. These activities included: meeting or discussion, attending or giving a presentation, phonecall, working on the computer (in the user's own office), taking a break, carrying out a "physical" activity around the office (e.g., add paper or get printer unclogged, do kitchen duty, etc.), and commuting. Respondents were asked to rate the relevance of each of these activities for setting their phone on "only lights" mode, silent, normal, or loud mode respectively. Additionally, the users had the option to input other relevant activities for each of these phone modes.

One could notice certain trends though there was no clear majority decision for any combination of phone mode and user activity. For instance, more than half of the users consider being in a meeting as very relevant for setting their phone on silent or "only lights" mode. For the other activities opinions are even more diverging, e.g. when taking a break from work.

Furthermore, only a subset of these afore-mentioned activities were deemed as relevant for at least one phone mode. Thus, making a phonecall, carrying out a physical activity around the office, and commuting were rated with an averge score of about 2 or less on a 1 to 5 scale.

Additionally, there were only four additional activities inputted by the respondents. Each of these activities was inputted by a single participant and could be seen as part of one of the original activities we have suggested.

To sum up, there is little agreement between users regarding the choice of phone mode for the different activities. The only agreement, albeit not unanimous, is that during meetings or presentations silent or "only lights" mode should be used. For this reason automated solutions for phone mode adaptation need not to focus on the predefined correlations between phonemode and user activity, but rather provide users with a simple way to customize them.

4.6 CONCLUSIONS

The frequency of manual phone mode adaptation by the users, together with the rate of occurrence of disturbances together with missed phonecalls, highlights the need for solutions to support the users with this issue. The phone's position and the user's activity can be useful factors to take into account when deciding the appropriate phone mode. More than 60% of the users are open to using such approaches, provided that they are designed in a privacy-conscious manner. Furthermore, users favor automatic phone mode adaptation over applications that would use notifications to remind them to adapt the phone mode. There is no consensus regarding the appropriate phone mode for the different situations, even for apparently obvious cases such as being in a meeting, therefore the best solution is to allow users to personalize these settings.

5

PHONE POSITION DETECTION

The way a phone is carried and stored has an impact on the the user's interaction with the phone and therefore the desired phone mode, as we have already shown in Chapter 4. For instance, if the user holds the phone in his hand, it is easy to capture his attention for an incoming phonecall, so adapting the display and triggering the vibration motor would be sufficient, a loud ringtone risks being unpleasant for the user, regardless of his activity. Furthermore, phone position can give us further insight into the user's activity and their willingness to be interrupted. Continuing the previous example, a user holding their phone during their meeting signals he is not completely engaged and perhaps there is a transition in the meeting, so he could be willing to be discreetly made aware of a new text message.

In this chapter we first define the set of phone positions we will be considering and then proceed to present our approaches to determine phone position. We first investigate single sensing modalities and determine the advantages and drawbacks of each sensor. As a solution to challenges regarding accuracy and noise resilience we propose several approaches using active probing [25, 26, 27]. Furthermore, in order to solve issues regarding the currentness of the last known phone position, energy consumption, and user disturbance, we present two approaches that use incoming notification sounds and vibrations opportunistically [25, 26]. Each of these aforementioned approaches are suitable only for a subset of the phone modes, for instance using incoming notification sounds as pilot sequences relies on the assumption that the phone is not on a silent or "only lights" mode, where notifications produce no sound. In the final step, we look at sensor fusion approaches that are applicable regardless of the phone mode.

5.1 DEFINITION OF THE CLASS SET FOR PHONE POSITION DETECTION

Let us first narrow down all possible environments or containers where a phone could be held or stored to the essential places that are being used most by the users in their daily activities.

Cui et al carried out a large-scale study investigating the ways users carry their phones [21]. While there were significant variations amongst culture, ethnic groups and genders, there was a limited number of phone positions that people used regularly. In general, users keep their phones in bags, pockets or belt clips, or in their hands. Thus, the first two positions represent storage spaces for the phone that usually involve the phone being carried around and thus being most likely in the close proximity of the user. We found it useful to add an additional position, namely desk or table, which would imply that the phone is stored on a surface which does not offer any guarantee regarding proximity of user: the user might be sitting at the desk the phone is lying or he might have simply left and left the phone behind.

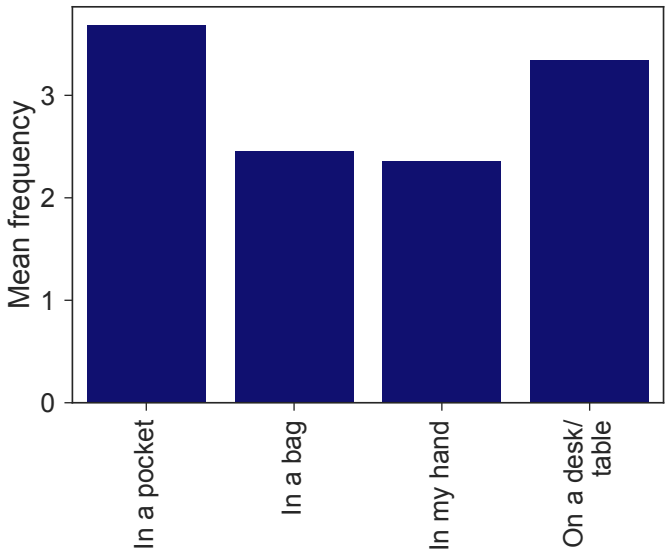
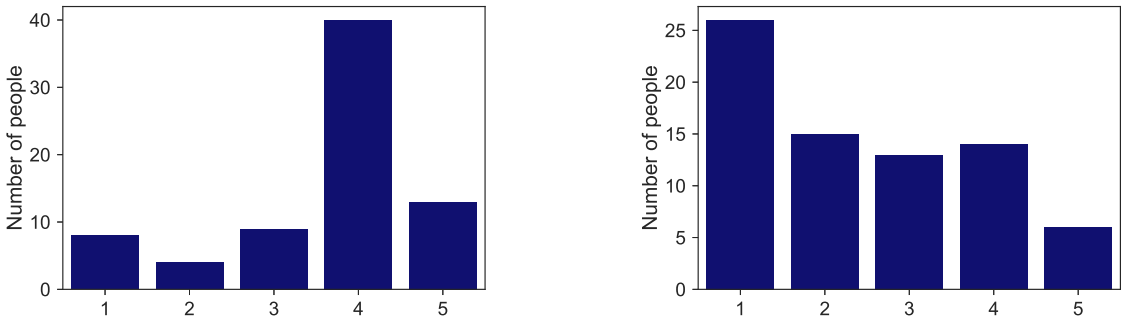


Figure 13: Mean frequency of the occurrence of the four phone positions

In order to further confirm these assumptions, we have investigated this aspect in our afore-presented user study. Thus, respondents were asked to rate the four phone positions in terms of how often they use them. They had five options, ranging from "never" to "all the time". Users had the possibility of manually adding up to three additional phone positions and rating their usage frequency. Figure 13 presents the mean values for the four considered phone positions, while Figure 14 provides a more detailed view by showing the distributions of frequencies for each phone position. While the relevance of the pocket and desk is clear, a bag or the user's hand seem to be of little relevance. This aspect can be clarified by looking at how the user's gender influences this results, as shown in Table 3. Unsurprisingly, men find bags most irrelevant, with a mean value of 1.83 compared to 3.52 for the women ($p<0.001$). It should be also noted that this is one of the most important phone positions for women, coming second only to desk, and being followed by pockets. In contrast to this, the majority of men prefer to wear their phones in their pockets, with a 4.04 mean value ($p=0.002$). For these reasons, detecting when the phone is in a bag is relevant for a significant part of our users and therefore the bag will be part of our class set.

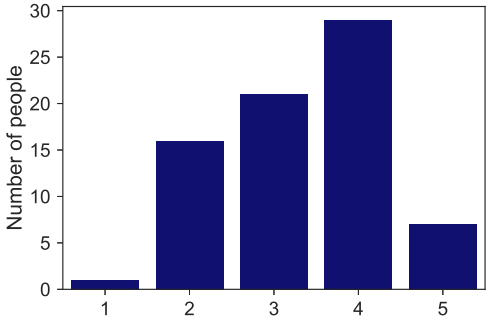
Despite respondents using hand as the least frequent phone position, it still plays an important role in deciding the optimal phone mode and making inferences about the user's behavior. Thus, knowing that the user carries a phone in his hand does not only tell us that any usage of loud ringtones would be unpleasant, but also gives us hints about the user's interruptibility and current activity. Additionally, 32 users have rated this position with 3 or more on the 1 to 5 scale. Therefore, we will consider the user's hand as part of our class set as well.

Lastly, we investigated whether there were additional important phone positions that we might have missed. As previously mentioned, users had the opportunity to mention up to three additional phone positions that they might be using. Only four respondents, which is 5% of the total number of participants, added one or two additional phone positions. None of the respondents used all three fields available

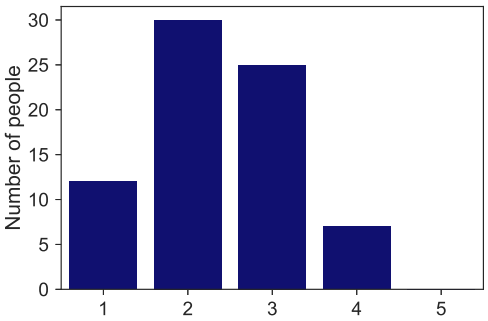


(a) Pocket

(b) Bag



(c) Table/Desk



(d) Hand

Figure 14: Distribution of the frequencies of the four phone positions

Phone position	Women (mean)	Men (mean)	t-value	Significance value
Pocket	3.04	4.04	t(33)=-3.38	p=0.002
Bag	3.52	1.83	t(46)=6.10	p<0.001
Desk	3.78	3.09	t(50)=3.05	p=0.004

Table 3: Impact of gender on preferred phone positions

for adding extra positions. Part of these additions do not represent phone positions, but rather user environments, like "at home", "in the other room" or "in the car", or an action that the phone is undergoing: "charging". There is a single suggestion for an actual phone position, which is "on the bed". We will not include this position in our class set given that only one user found it relevant and since the ideal phone mode in this case would be quite similar to that when having the phone on a table while at home.

To sum it up, our user study confirmed and reinforced the four phone positions we proposed: pocket, bag, hand, and table/desk. It highlighted not only the importance of these positions but also the fact that there are no other additional phone positions that the users we interviewed find important and that they use regularly.

This four phone positions can be split into two categories: "inside" and "outside". Thus, pocket and bag represent enclosure or "inside" spaces for the phone, while hand and pocket are open spaces, or "outside". We will be using this categorization of phone positions in the following sections.

5.2 PASSIVE PROBING APPROACHES TO PHONE POSITION DETECTION

First attempts to determine phone position are based on collecting and classifying data from smartphone-embedded sensors. The usage of two specific sensors, the microphone [87] and accelerometer [38, 129, 158], were already proposed by related work, however with different scope and class sets. For this reason, we needed to evaluate these methods and determine their appropriateness for our own class sets. We discuss these issues in more detail in the subsections corresponding to these sensors. On the other hand, the approaches using certain sensors for classifying where the phone is "inside" or "outside" of an enclosure are our own contributions.

As previously mentioned, we only consider hardware sensors in order to make our application available to the widest array of users, regardless of whether they use certain mobile applications.

We can split smartphone hardware sensors into three categories: sensors that can be used as single sensing modality for phone position detection, sensors that can provide extra insight for uncertain situations, e.g., indicate if the phone is inside or outside of an enclosure, but cannot be used on their own for determining phone position, and sensors that cannot serve us in any way for the given task.

5.2.1 *Determining Phone Position Based on Single Sensing Modality*

Using the microphone or accelerometer to classify phone position has been proposed by the related work [38, 87, 129, 158], however with different class sets, which in some cases are quite limited. For instance, Miluzzo et al [87] only determine whether the phone is inside or outside of an enclosure. For this reason, we evaluated these methods on our own class sets.

Audio recordings of environment sounds alone provide acceptable results, especially in silent environments. This stems from the fact that the audio signal propagates differently based on the place where the phone is held. Such patterns stand out particularly when playing a sound and recording it in the same phone position. However patterns can be identified also when recordings environment sounds. Thus,

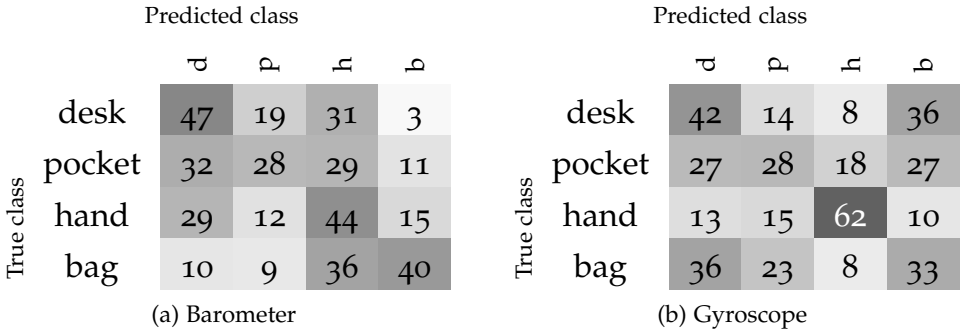
sounds will be muffled in a pocket or bag, more so for the former and less in the latter. The closest recording to the environment sounds themselves will be obtained when the phone is in the user's hand. Miluzzo et al. [87] used this method to classify whether the phone is inside or outside of an enclosure, obtaining accuracies of up to 77-84%. Furthermore, phone position itself plays a significant role in classification accuracy, pocket and bag being the classes with the greatest accuracy. Thus, there is an obvious improvement potential with regards to overall accuracy and impact of noise.

Also accelerometer readings can be used to infer phone position. Thus, phone movement will be affected by the phone position. A phone on a table will generally be stationary, while a phone in a pocket or hand will display quite distinct movement patterns. A bag can be either stationary or in movement, depending whether the user is carrying or storing it. Thus, a stationary bag and a desk are almost impossible to distinguish using only passive probing. This method is obviously unaffected by ambient noise and our preliminary tests have shown that impact of user movement on classification results are quite limited. However, there is still quite some potential for classification accuracy improvement. On a different class set, Wiese et al [158] report accuracies of up to 85.7%.

5.2.2 *Sensors That Are Unfit For Phone Position Classification*

In what follows we shall discuss the potential role that the other sensors embedded in LG Nexus 5 phones can play in phone position classification. Nexus 5 was one of the state-of-the-art smartphone models at the time of the evaluation, having a wide array of sensors with the highest sampling rates out of all phone models we have considered. At the same time, we did not consider sensors that were included only in very few smartphone models, such as temperature and humidity sensors, in our attempt to make our approaches applicable to as many users as possible.

Thus, aside from the microphone and accelerometer, experiments have shown that none of the other sensors incorporated in LG Nexus 5 phones can be used as single sensing modality for phone position classification. All these other sensors can either be used to determine only a very coarse-grained position like "inside" or "outside" of an enclosure, or are simply unsuitable for the classification task. Sensors that can be used to determine hints include the light sensor and the proximity sensor, which we will be analyzing in more detail in the following section.



(a) Barometer (b) Gyroscope

Figure 15: Confusion matrices for barometer and gyroscope sensor data (in %)

Sensors that are inadequate for phone position classification include GPS, Wi-Fi [SSID], barometer, compass, gravity, gyroscope, magnetometer, and pressure sensor. Thus, our preliminary test results have shown that these sensors do not achieve better results than randomly selecting the phone positions. This finding is quite intuitive since all these sensors measure environment conditions which are very little, if at all, affected by phone position. User location provided by GPS or Wi-Fi traces cannot provide any indication regarding phone position. While there might be some faint variation in the signal strength, this is not enough to distinguish between the phone being on a table or in a pocket. Similarly, air pressure is not influenced by phone position, Figure 15a showing that classification results are even worse than the random selection of a phone position. The gyroscope traces do not exhibit any recognizable patterns in most phone positions. Thus, except for the hand position, the rotation movement is minimal in all other positions, the roll, pitch and yaw values being very small. Figure 15b shows that only when the phone is in the user's hand classification accuracy is higher than 50%.

5.2.3 *Using Smartphone Sensors to Determine Coarse-Grained Positions: Inside vs Outside of an Enclosure*

Sensors that can only be used to classify coarse-grained positions and thus can only be used as extra validation for the classification result of other sensors, are sensors whose data is influenced in a similar way by multiple phone positions. For instance, the light or proximity sensor are not able to distinguish whether the phone is in a pocket or bag. Therefore, such sensors are unfit for phone position classification, but can still be used to determine whether the phone is inside or outside of an enclosure. Additionally, they can provide hints and help enhance classification accuracy in certain cases. For instance, when classifying multiple data windows from a certain sensor or data from multiple sensors, like microphone and accelerometer there is sometimes a disagreement about the classified class. If the number of votes for two different classes are very close and the two positions are one "inside" and one "outside" the light or proximity sensor could be used to determine the actual class.

We carried out a preliminary evaluation of the results obtained by light sensors readings in distinguishing "inside" from "outside" positions. The data was collected using LG Nexus 5 phones. Classification results for "inside" vs "outside" positions achieve almost 100% precision and recall for the light sensor. These values are not affected by ambient light, except for pitch-black spaces. It would be impossible to distinguish whether the phone is on a desk or in the user's hand in a dark space, or simply in the user's pocket in a normally lit space. Therefore, during the night when the user sleeps, the light sensor would be seriously challenged. One could argue that such situations could be specifically handled by the activity classification module, since when the user is in a dark room it is either for sleeping or other activities when he would not want to be disturbed.

As far as the phone's proximity sensor is concerned, it has achieved 100% precision and recall in our preliminary tests. The tests were carried out using the same phone model, LG Nexus 5. The main drawback of the proximity sensor is it provides readings only when there is a state change which makes it unsuitable for short recordings.

5.3 IMPROVING PHONE POSITION DETECTION THROUGH ACTIVE PROBING

In what follows, we describe our contributions concerning phone position classification through the usage of pilot sequences. Our main aim is to improve classification accuracy in all situations, including noisy ones, while causing no or ignorable disturbance to the user. Further challenges that we tackled were gathering sensor data in a way that ensures that the samples correspond to the current phone position without draining the phone battery or creating significant privacy concerns. Thus, the former aim excludes duty cycling, while the latter dismisses continuous sampling. Last but not least, we aimed to offer more privacy-friendly alternatives to the users, for instance by relying on other sensors besides the microphone, since even a few hundred milliseconds of recordings might be problematic when it comes to sensitive situations.

5.3.1 *Using Audio Signals as Pilot Sequences*

The following approach relies on the assumption that sound is attenuated differently for the most common phone positions. For instance, sound is muffled in a pocket or a bag, more in the former and less in the latter. This stands true to a certain extent for recordings of environment sounds alone, however classification results obtained this way range between 77 and 84% accuracy. Additionally, results are quite affected by environment noises and by the actual position of the phone itself, accuracies being lower in certain phone positions. For instance, the pocket and the user's hand are often confused.

In order to overcome these issues, we probe the attenuation of a deterministic signal in the phone's environment. This is aimed to make the different phone positions have more distinct fingerprints and thus improve classification accuracy. Thus, instead of classifying recordings of environment sounds alone we classify recordings of an audio signal we are playing and of the inevitable environment sounds. Furthermore, we wanted to investigate whether pilot sequences could help us distinguish whether the phone was lying normally on a table, facing the ceiling, or was facing the table. This detail could be useful in certain situations when we might want to visually notify the user of an important message. This approach was already published, the current section being based on [27].

Figure 16 highlights the acoustic fingerprints of the different phone positions by showing side by side the spectra of recordings taken in all phone positions, while using Gaussian noise as pilot sequences. As we can see in the figure, sound is muffled in the "inside" locations like pocket or bag. However, the two positions are still distinguishable since in general pockets are tighter than the inside of a bag and thus the sound is more pronouncedly muffled. On the other hand, the two "outside" positions, desk and hand, present quite different patterns. The sound recorded in the user's hand is most similar to the originally played sound. When gathering the recording on a desk however, the sound is amplified because the table acts as a resonating body.

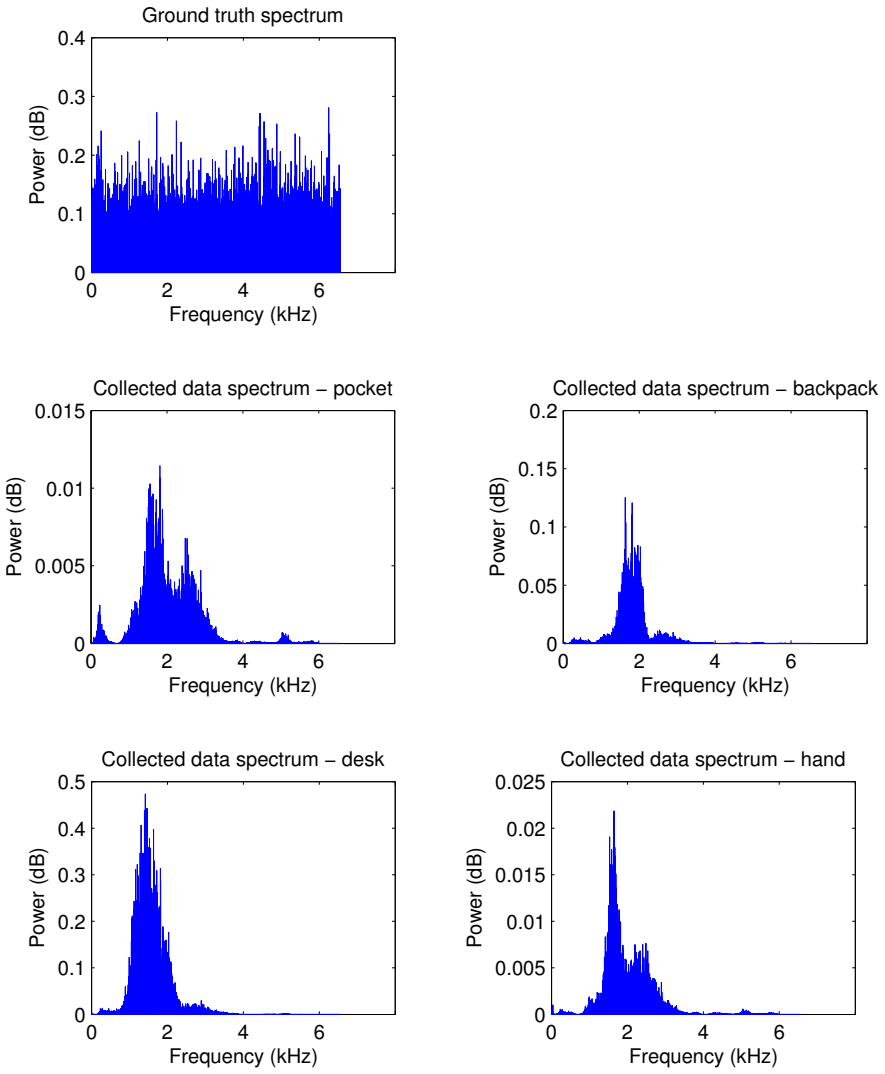


Figure 16: The spectrum of the Gaussian noise sample, as well as of the samples collected with a phone located in the pocket, backpack, on the desk and in the user's hand

5.3.1.1 *Concept*

Our concept is to use the same phone to play a pilot sequence and record it concomitantly. This process will be repeated at fixed intervals. While this presents the risk of not knowing the most recent phone position, continuous recording is also unfeasible due to the obvious user disturbance, battery draining and privacy concerns.

Our overall system, as shown in Figure 17, consists of two main parts: a mobile application that gathers the data and a server-side application that classifies the collected samples.

The mobile application periodically plays certain pilot sequences and records them at the same time. The volume of the sound snippet we play is adjusted for each recording cycle. Thus we first of all aim to avoid clipping. However, a too low volume of the played sequences has a negative impact on the classification results as well, since recordings collected this way have less pronounced fingerprints of the different phone positions. We have experimented with four different pilot sequences. The first

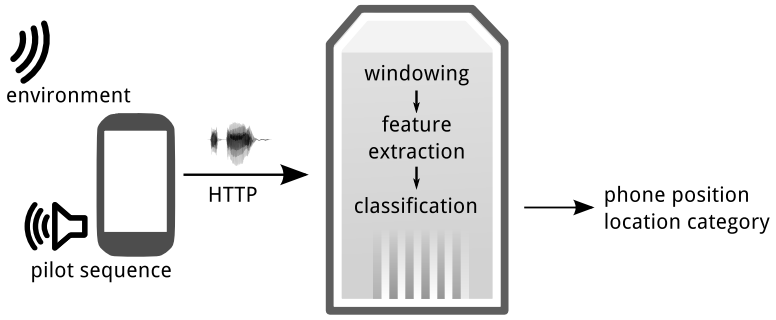


Figure 17: System architecture

of them was Gaussian noise as a general way to evaluate our approach. We further selected a few sequences composed of prime numbers, such that harmonics ranging at multiples of the fundamental frequency will not impact our results. The audio data we intended to collect was to be sampled at 44.1 kHz, so due to the Nyquist limit, we selected only frequencies under 20 kHz for our probing sequences. We used two sequences of pseudo-logarithmically distributed primes, half of them distributed under 1 kHz and the other in the 1-20 kHz range. The first of the samples uses 40 primes, while the second uses 316, including all 158 primes under 1000. A further pilot sequence we used included 40 linearly distributed primes under 20 kHz.

On the server side, the recordings are classified using a custom-made processing pipeline. We describe this process in more detail in the following section.

5.3.1.2 *Processing and Classification of Samples*

The recordings are classified using a custom-made processing pipeline. As shown in Figure 18, it consists of five main stages: windowing, silence removal, Fourier transformation, feature extraction and classification. Thus, our approach relies mainly on machine learning techniques for solving the phone position classification problem, while leveraging some signal processing methods to filter out silence and obtain the frequency domain representation of the signal.

In the first step, if the recording length is longer than a certain value, the recorded signal $s(t_n)$ is split into equally sized windows, $w(t_n)$, each containing 4096 samples. At a sampling rate of 44.1 kHz, this gives us a window size of 92 ms. This value varies has been determined experimentally. The intuition behind this decision is the following: Recordings under a certain size do not contain enough information to allow an accurate classification. At the same time, gradually increasing the window size past the optimal length at the beginning does not lead to an increase in classification accuracy and afterwards leads to a gradual decrease in accuracy due to overfitting. Additionally, longer recordings are more likely to disturb the user.

It is worth noting that since the environment influence on the ground truth signal is constant (e.g., muffling), one can use very small and constant-sized windows for phone position classification. Shorter recordings imply less risk of disturbance and privacy concerns for the users. As a counter example, classifying keys pressed by a user requires that each fragment corresponds to one keystroke, from starting to press the key to fully releasing it. This leads to variable lengths and thus requires a more sophisticated segmentation.

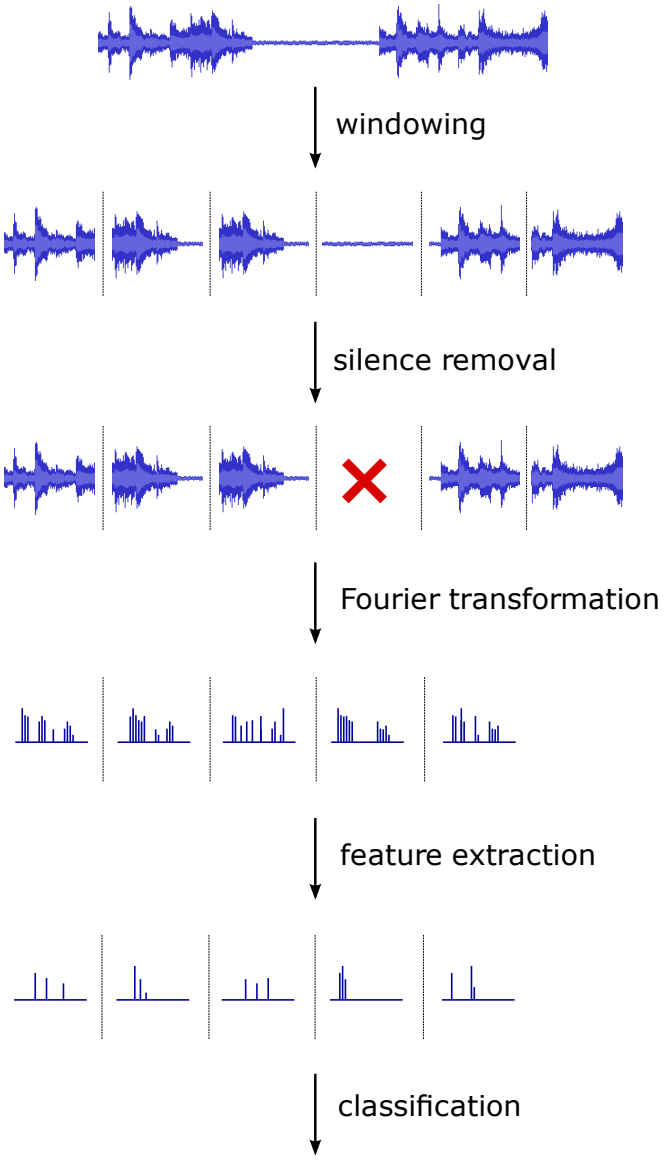


Figure 18: Classification pipeline

After splitting the original recording in multiple windows, or immediately after receiving the data if the recording has the required size, we apply a windowing function. Our approach allows the usage of rectangular, Hamming or Hanning windows, and uses a configurable overlap between the windows. Preliminary results have shown rectangular windows to be associated with better classification results, thus those are the default used by our application.

In the next step, silence removal, we discard windows that do not contain sufficient information and thus would lead to incorrect classification results. This step is required by most of our approaches, and not just for the baseline classification of environment sounds. In the current scenario there is little chance of having silent windows when playing pilot sequences, since the recording aims to completely overlap with playing the signal. However, in the upcoming approaches this is an essential step. For instance, let us assume we record ringtone sounds. In this case there will

be silent windows in between the repetitions of the ringtone. Not removing those windows would lead in the best case scenario to results as good as that of the passive probing approach. Therefore, we compute the signal energy for each window. If it is under a certain experimentally-established threshold, the window is discarded, otherwise it is fed/forwarded to the next step of the pipeline.

In the first stage a Fast Fourier Transform is applied to the each window, transitioning from time to frequency domain. Since window length directly influences the frequency resolution of the Fourier transformation, selecting an appropriate window size is important for the overall system. Longer windows take longer to record and run a higher risk for user disturbance and privacy concerns, but they also lead to a higher frequency resolution. On the other hand, using smaller windows results in a lower frequency resolution but cause a lower computation complexity and less disturbance concerns. In our setup with a sampling rate of 44,100 Hz and a window size of 4,096 samples this results in 4,096 spectral components ranging from 0 Hz to 22,050 Hz with a frequency resolution of 10.77 Hz. However, since in the next step we run a dimension reduction operation by applying a feature extraction algorithm, we can accept the trade-off brought by smaller window sizes. Thus, we are interested mainly in the patterns existing in the data, which can be retained even with lower frequency resolutions. Since we compute a very reduced number of features (<20) out of the 4096 spectral components, the impact of frequency resolution is somewhat limited; computing it out of 8192 or 16384 samples would bring little improvement to accuracy, as our preliminary tests have shown. Using significantly smaller windows is however not feasible due to limitations posed by the phone operating system, in our case Android for synchronization of playing and recording the pilot sequence. Thus the internal scheduling of tasks causes variations which could lead to the recording and sound playing not to overlap at all.

Next come the most important elements of the processing pipeline. Here we leverage machine learning techniques to first reduce dimensionality of the data and then classify the results. In the fourth step we reduce dimensionality of the data by applying different feature extraction algorithms. We are thus trying to extract the patterns of the data and eliminate redundant dimensions in order to facilitate the classification of the data. We have considered several feature extraction algorithms, ranging from simple ones like computing the power spectrum or band energy, to the more sophisticated algorithms that have been developed mainly for speech recognition tasks but have proven to be successful for environment sound classification as well: Mel Frequency Cepstral Coefficients (MFFC) [73] and Delta Mel Frequency Cepstral Coefficients (DMFCC). We initially considered Zero-Crossing-Rate given its success in voice detection systems [69, 116]. However, it did not perform well in our scenario, presumably because it is applied in the time domain while phone position patterns are more distinguishable in the frequency domain, so after a series of tests we decided to discard it. MFCC are "perceptually motivated" [150], emulating the behavior of the human auditory system [141]. MFCC has been used in the past for speech [92, 151] and speaker recognition [141], but also for general audio classification tasks [97, 123]. Similarly to [97, 105, 141], we used the first N=13 MFCC coefficients. One potential shortcoming of MFCC, depending on the classification task is that it takes no temporal aspect into account, the features being calculated for individual windows. As a solution to this, DMFCC appends to the MFCC coefficients the first order derivative of the MFCC coefficients with respect to time. Thus, DMFCC

has 2*N coefficients, which depending on the scenario and evaluation settings can be a drawback which overwhelms the advantage brought by taking into account the temporal aspect. This issue is discussed in Section 5.3.1.3.

In the last step, the previously computed features are classified in order to determine the phone position. Our framework supports the usage of arbitrary classification algorithms. However, regardless of the type of features we have selected in the previous step, the input for the classification phase will be N numerical values which express the patterns in the distribution (spectral shape) of the signal's frequency components.

We considered only supervised learning methods, since we have a known set of possible classes and do not aim to discover unknown patterns in the data. Obtaining labeled data is often a time-costly process, requiring manual input from users. Unlabeled data is much easier to obtain, however our goal was to detect certain phone position and not identify clusters in the data based on signal attenuation patterns. While to a certain extent there might have been an overlap between the clusters and our initial classes our main goal was nevertheless the recognition of those specific classes, i.e. phone positions. In between supervised and unsupervised approaches lie semi-supervised methods, which identify patterns in the data then make use of a lower number of labeled samples in order to train models that can be used for classification. However, for our application scenario, which deals with user disturbance, classification performance is essential. Therefore, we have decided against semi-supervised methods despite the increased cost of data collection that this decision involved. The choice of specific classifiers was based on examining related works with comparable classification problems and datasets. The choices were narrowed down by using Occam's razor principle, i.e. starting with the simpler approaches and proceeding only until results were very good. We thus used Naïve Bayes (NB), Decision Trees (DT), Random Forest (RF), Gaussian Naïve Bayes (GNB), similarly to [87], K-Nearest Neighbors (KNN), and Gaussian Mixture Model (GMM), like [87, 107], and Support Vector Machine Classifier (SVM).

5.3.1.3 *Evaluation*

We focus in what follows, as well as in the remainder of this chapter, only on assessing the classification performance, leaving aside other aspects like energy efficiency. Issues of practical realization and feasibility, such as in-time classification, its relationship with user disturbance, and its consequences on the selection of sensing modalities are discussed in Chapter 6.

For the evaluation we used Samsung Galaxy Nexus and Samsung Galaxy S3 phones, being chosen for their audio recording quality and their support of a sampling rate of 44.1 kHz. We had two users for collecting the training samples and carrying out the evaluation. They manually labeled the upcoming sample, then started the recording. Our application dismissed the first few seconds of all recordings since the user had to label the phone position while holding the phone most likely in his hand and had to afterwards move it to the target position according to the inputted label. At the end he stopped the recording. This lead to the creation of a good quality dataset, that required minimal cleaning, e.g. removing very long recordings where it appears that the user forgot to stop the recording. We collected a total of 7862 samples while playing Gaussian noise and a similar number of environment sounds

recordings for comparison purposes. For each of the other pilot sequences, we gathered on average 1640 recordings. The classification accuracy of our prediction model has been evaluated by means of a 10-fold cross validation, where 90% of the data is used for training and 10% for evaluating the results.

While 92 ms recordings are sufficient for our system to work properly, for training and evaluation purposes we used longer recordings, and thus can ponder on the disturbance of the sequences themselves. The samples were played at less than half of the maximum volume supported by the phones in order to avoid clipping. Therefore, the noise was quite easy to ignore or even go unnoticed in noisy environments like outdoors or in a bus, and even in silent environments if the phone was in a backpack.

Since our aim was to develop a method that achieves high accuracy even in less optimal situations, we evaluated our approach in a variety of situations. Thus, we gathered and classified data in both silent and noisy environments, as well as in situations where the user or his environment were in motion or stationary. Table 4 shows the environments in which we evaluated our approach. As previously mentioned in 5.3.1, we also investigated whether in the case of a phone lying on a table we can distinguish between the cases when it is facing the desk and when it is facing the ceiling.

Our initial plan was to label also the user's environment in order to establish whether there are significant differences in classification performance and thus by the determining the most problematic environments to get insights into what harms the classification accuracy. However, given that knowledge about these environments could influence the decision of which sensor to use in future extensions of this work and what confidence to assign to each sensor, we used the audio recordings to classify also the type of environment. Additionally, the type of environment that the user is currently in could be used to provide more insight into what the appropriate phone mode would be. We present these results together with phone position classification results for each of the pilot sequences.

Let us first analyze the classification performance for samples collected while generating Gaussian probing sequences. Figure 19 shows the accuracies for all the combinations of features and classifiers that we have considered. As previously mentioned, we also use accuracy as a metric in order to facilitate comparisons with the related work. In this case, our baseline is the approach of Miluzzo at al [87] which records environment sounds alone and aims to determine whether the phone is inside or outside of an enclosure, without differentiating between more specific phone position. Their approach achieves accuracies of up to 84%.

As can be seen in Figure 19, our approach obtains accuracies of up to 96.99% for the MFCC and K-Nearest Neighbours combination. As far as feature types are concerned, MFCC and Delta MFCC perform the best, MFCC results being slightly better than those obtained by Delta MFCC. This happens due to the fact that Delta MFCC has double the number of coefficients of MFCC, which significantly increases the complexity of the classification problem, while at the same time not benefiting much from the temporal aspect being taken into account, since the effects of phone-position on the sensor readings are constant in time. Thus, we suspect the Delta MFCC slightly overfits.

K-Nearest Neighbors and Random Forest are the classifiers that yield the best results. Decision Trees have a lower accuracy likely because the data is too complex to be properly modeled by a single tree. Thus, while not displayed here, our re-

Code	Phone position	User environment	State	# Windows
bb	Backpack	Bus	Motion	3975
bo	Backpack	Office	Stationary	11867
bos	Backpack	Outdoors	Stationary	4015
bow	Backpack	Outdoors	Motion	3699
bt	Backpack	Tram	Motion	3978
ddof	Desk (down)	Office	Stationary	11952
ddou	Desk (down)	Outdoors	Stationary	4028
duof	Desk (up)	Office	Stationary	11818
duou	Desk (up)	Outdoors	Stationary	3975
hb	Hand	Bus	Motion	3780
ho	Hand	Office	Stationary	11685
hos	Hand	Outdoors	Stationary	4011
how	Hand	Outdoors	Motion	4024
ht	Hand	Tram	Motion	4016
pb	Pocket	Bus	Motion	3926
pofs	Pocket	Office	Stationary	11960
pofw	Pocket	Office	Motion	11931
pous	Pocket	Outdoors	Stationary	3991
pouw	Pocket	Outdoors	Motion	6336
ptrai	Pocket	Train	Motion	12024
ptram	Pocket	Tram	Motion	11973
			Total no. of windows	148964

Table 4: Phone positions and user environments used for the classification

sults showed that in-sample errors of Decision Trees were also higher than those of K-Nearest Neighbors and Random Forest. The Support Vector Machines algorithm proved the most inappropriate for our scenario, since the training process did not converge due to the high amount of data. Thus we had to significantly reduce the amount of data to use for training in order for it to converge, which in turn lead to poor classification results. The types of feature vectors used had varying impacts on classification accuracy. This impact was most noticeable for the Gaussian Mixture Model, where average classification accuracies range from 90.01% and 89% for MFCC and Delta MFCC yield an average accuracy of 90.01% and 89% respectively to 59% for band energy.

In what follows, we will closely compare the results yielded by samples containing just environment sounds and by samples where Gaussian noise was used as pilot sequence. We classify both the phone position and the user's environment (cf. Table 4). Since MFCC and K-Nearest Neighbors obtained the best results for both types of recordings, we are going to use this combination for the performance comparison. Figures 20 and 21 presents the confusion matrices for the two types of recordings, highlighting a clear improvement brought by Gaussian noise. Thus, precision and

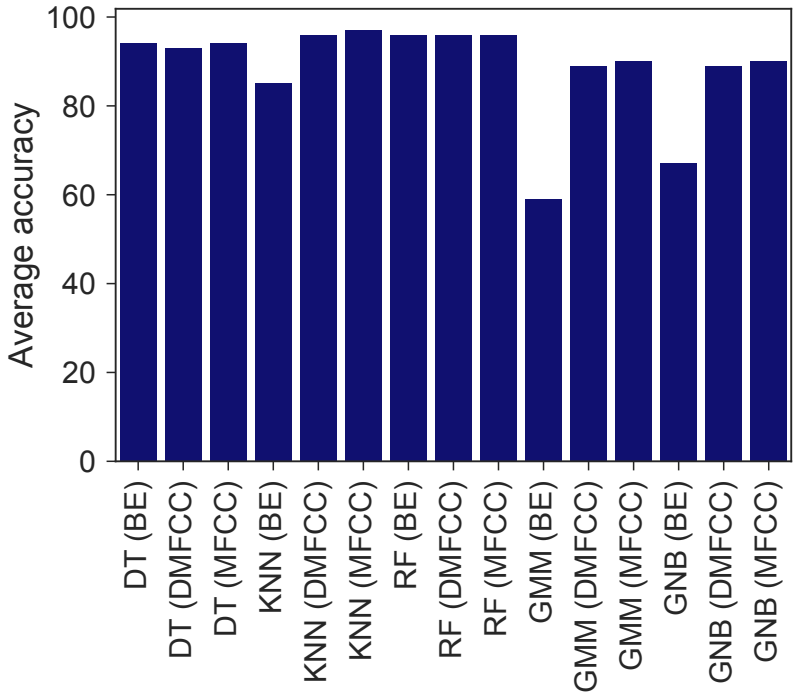


Figure 19: Comparison of the classifier performances and feature types

recall increase from 71.76% and 70.22% to 93.22% and 96.77% respectively. One can notice that in the case of environment sounds recordings it is more difficult to distinguish between somewhat similar phone positions, e.g., between a tight pocket or a big compartment of a backpack, or, even more so, between a phone lying on a desk facing the ceiling or the desk. Recordings of environment sounds alone face an additional challenge when it comes to distinguishing between the different environments, even if they exhibit different noise levels (e.g., indoors vs. outdoors).

The usage of Gaussian noise as pilot sequence leads to very good results both for phone position and user environment classification. For the former, this confirms our initial assumption that the sound propagation patterns are very pronounced when playing a pilot sequence, in this case Gaussian noise, being much more obvious than those of environment sounds alone. For the type of environment classification it seems that Gaussian noise helps cancel out some of the irrelevant environment noise, thus making environment-specific features much more obvious as a difference between the frequencies of the recordings and those of the initial noise.

If we look in more detail at the false positives and false negatives rates for the individual classes when using Gaussian noise, we can notice that a significant part of the misclassifications happen because of the type of the environment and not phone position. For instance, on average, 3.6% of the cases when the phone was in the user's pocket in a tram were misclassified as the phone being in the user's pocket in a train. Since this is one of the highest false positive rates for an individual class, it is worth noting that the two environments, train and tram, have quite similar acoustic pattern so they're much more difficult to distinguish than other types of environments.

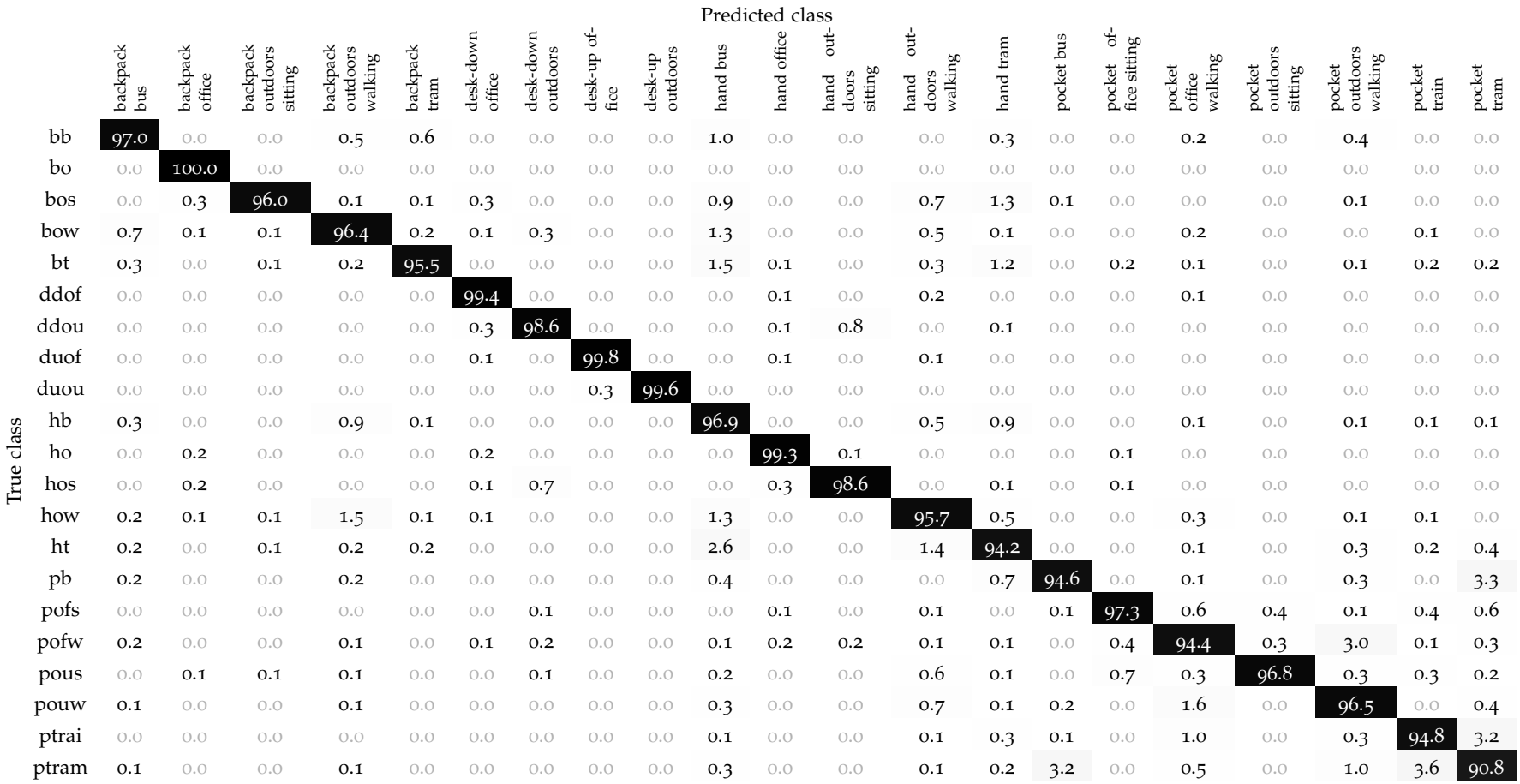


Figure 20: Confusion matrix for recordings with Gaussian noise (148964 windows, results in %)

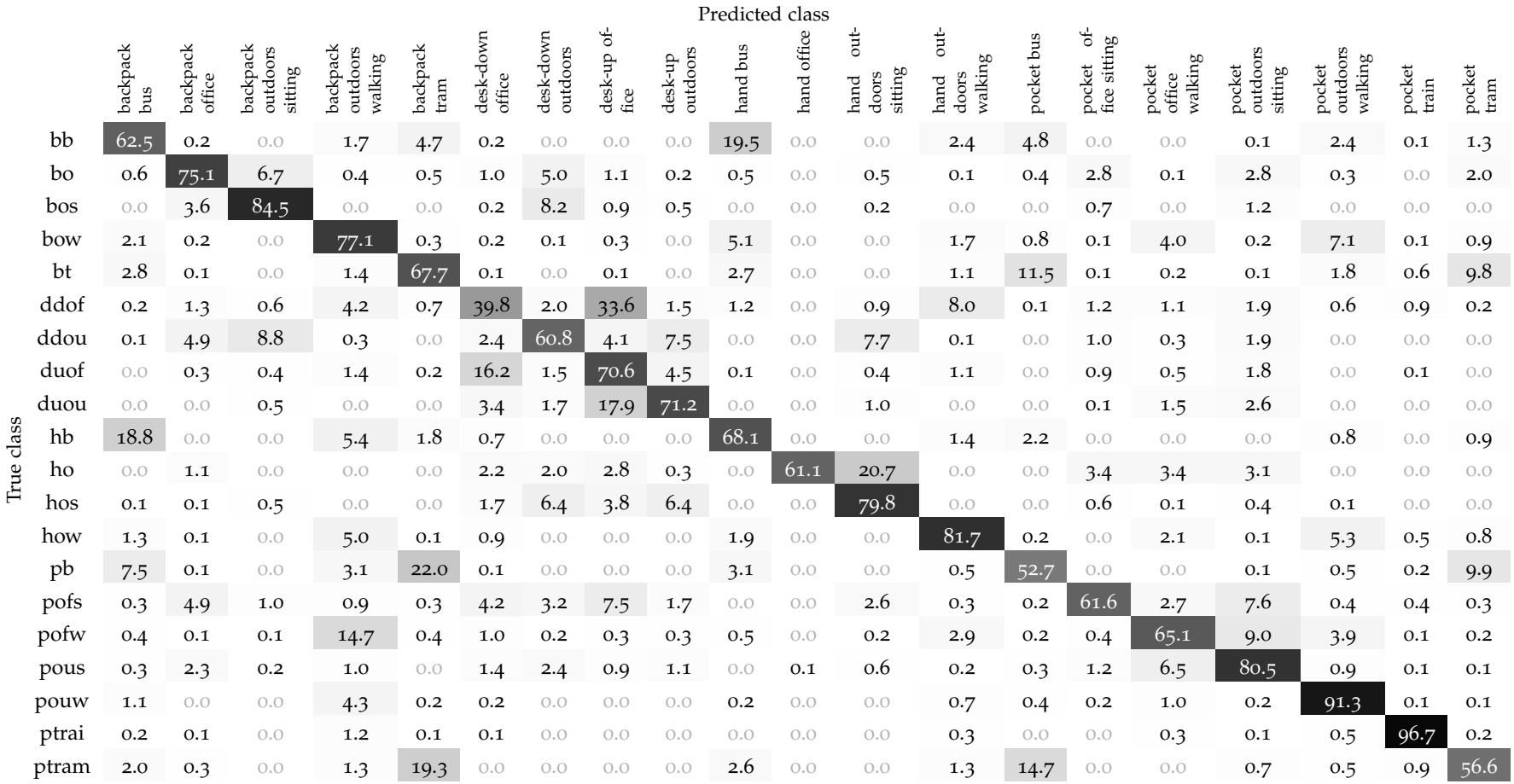
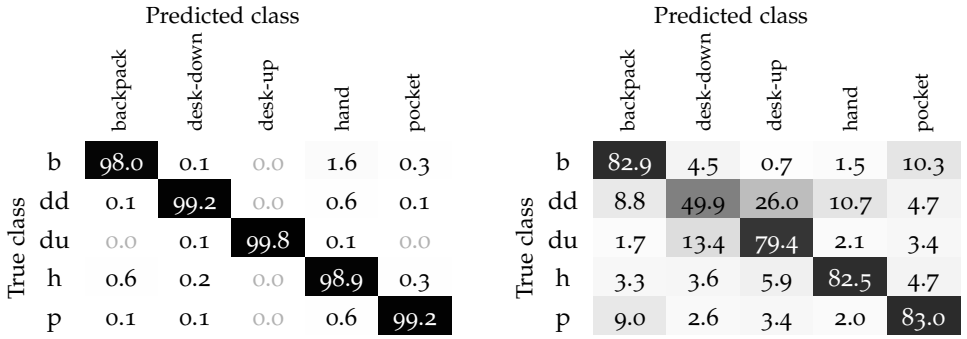


Figure 21: Confusion matrix for recordings without pilot sequence (82296 windows, results in %)

Next we will focus on the classification results for phone position alone. We use the same dataset for the evaluation in order to preserve the mixture of situations and noise levels, but simply ignore the type of environment. Figures 22a and 22b present the confusion matrices corresponding to the two approaches: recordings using Gaussian noise as pilot sequence and recordings of environment sounds alone. Similarly to the previous comparison, we used K-Nearest Neighbors and MFCC for the classification.



(a) Gaussian noise (148964 windows) (b) No pilot sequence (64826 windows)

Figure 22: Classification results for the phone positions (in %)

The already excellent results achieved by Gaussian noise see an improvement from 93.22% and 96.77% precision and recall respectively to 98.90% and 99.01% while results for environment sounds alone increase from 71.76% and 70.22% to 77%. The improvement is not surprising, since we reduce the problem complexity by classifying only phone position and not user environment alike. As already mentioned, a significant percentage of confusions in the previous classification problem involved just misclassifying the type of environment, while correctly determining the phone position. Differentiating between the two desk positions based on the side the phone is facing towards makes an insignificant difference, the precision and recall obtained when classifying only the desk position as a whole being 98.76% and 98.91%. The results achieved by Gaussian noise however clearly argue for suitability for such a task, given the variety of environments they were evaluated on. There is a certain degree of variation with regards to noise level and acoustic patterns even within the same environment. For instance, in a tram some recordings will have just tram sounds, some will have the tram stopped in a station, some will have people talking and others will have the next station announced on the speakers. Therefore, it is important to note the robustness of the method under the given conditions.

Additionally to Gaussian noise, we have experimented with a number of semilogarithmic series of primes. They yielded similar results to Gaussian noise in silent environments, while having sensibly worse results in noisy settings. It should be noted that all sequences of primes have similar results which come very close to a 100% precision and recall silent situations. Additionally, the sequences of primes obtain better results in some specific cases, for instance in the case of a phone lying on a desk facing the ceiling in the office, where a semilogarithmic sequence of 40 primes classifies it correctly in all cases for the given data set. The Gaussian noise signal has the best overall performance, as it is less affected by noise than the other

signals. However, these results were on average still better than those of recordings of environment sounds only.

5.3.1.4 *Reached and Remaining Goals*

The afore-presented evaluation results clearly argue for the suitability of using pilot sequences in general and Gaussian noise in particular for phone mode classification, as well as for classifying a variety of environments the user can be in. This approach achieves excellent results in silent and noisy situations alike, having proven its robustness in a variety of daily situations the user may encounter. The almost perfect results achieved by phone position classification (98.76% and 98.91% precision and recall respectively) are a significant improvement compared to the passive probing approach, more specifically a 28.24% improvement.

However, there are still a number of open issues. First of all, despite using very small sample sizes, there is a chance of disturbance, especially in very silent settings where the short pilot sequence could still be perceived by the user. Secondly, this approach uses duty cycling, which cannot guarantee that the phone position did not change since the last recording. Furthermore, gathering recordings with a high frequency will drain the battery faster and bring an even higher likeliness of disturbing the user through the repeating sound snippets.

5.3.2 *Piggybacking Phone Notification Sounds*

Playing out sounds for the sole purpose of improving classification results presents the risk of disturbing the user. An easy solution to the user disturbance problem could be to use ultrasounds instead of the proposed pilot sequences. However, since current smartphones do not support neither playing nor recording ultrasounds, this approach is not feasible at the moment.

For this reason, we investigated the sounds that are played by the phone independent of our application and whether those can be used as pilot sequences. There are two main reasons for the phone to play an audio clip: the user listening to music or an incoming phonecall/message/notification. The former source is unreliable since it is limited to certain often irregular times and to only certain user activities, so it cannot be relied on for continuous context monitoring. Furthermore, sounds are often played on a headset. Sounds triggered by phonecalls and other notifications are also happening at irregular times and with uneven patterns. However, our goal is to adapt the phone mode so that the user does not face disturbance or unnecessarily missed phonecalls.

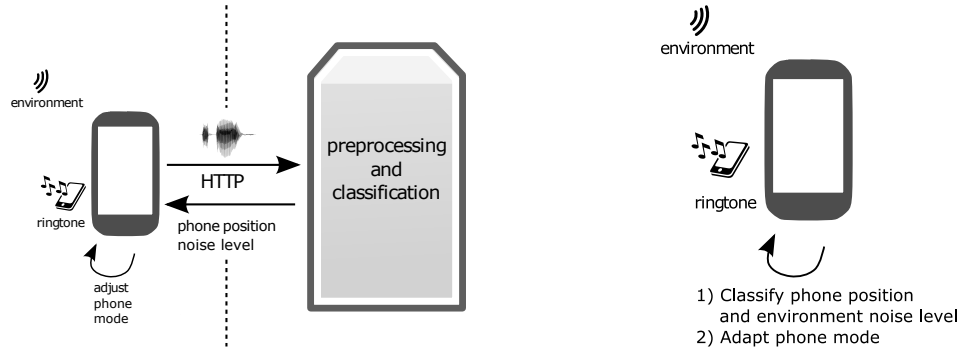
5.3.2.1 *Concept*

Our idea is to wait for incoming phonecalls, messages, and other notifications, then use the corresponding sound that gets played as pilot sequence. Thus, we are using the phonecalls/notifications/alarms in two ways. We use the sound associated with these events as pilot sequence, while at the same time using the event itself as trigger for our classification process. This has two big advantages: 1) the sounds are "free" and 2) we are aware of the exact moment when the phone mode needs to be adapted. The second advantage can help us avoid duty cycling and thus its drawbacks regard-

ing increased energy consumption and privacy concerns, and at the same time the risk of the phone position to have changed in the mean time. Additionally to phone position we also classify whether the phone is in a silent or noisy environment, since in certain circumstances this can be a factor that influences the choice of phone mode.

This approach presents two main challenges. First of all, we have to minimize classification as well as overall phone mode adaptation time in order to minimize disturbance potential. This issue is discussed in more depth in Chapter 6. The second question is whether ringtones and other sounds are good enough choices for pilot sequences, i.e., lead to a better overall classification accuracy and reduce the impact of noise compared to recordings of environment sounds alone. We explore these problems in Section 5.3.2.2.

We have considered two architectures for our approach, depicted in Figures 23a and 23b respectively. Thus, a mobile application listens for incoming events corresponding to phonecalls, notification messages, and alarms. When such an event happens, the application records a short fragment of the corresponding sound, together with the other environment sounds. In the case of the first architecture, **PCD-S** (Phone Context Detection – Server) this recording is sent to a server, where it undergoes preprocessing, feature extraction, and classification. The classification result is sent back to the phone and based on this the phone mode is adapted accordingly. The second architecture, **PCD-P** (Phone Context Detection – Phone), handles the whole processing pipeline on the phone, the phone mode being directly adapted based on the classification result.



(a) Server-based classification

(b) Phone-based classification

Figure 23: Architecture alternatives for data collection and phone position classification

We have decided to explore the two different architectures in order to determine experimentally which one is the most suitable for our problem. Both of them are frequently used in the related work. Doing all processing on the phone does not rely on network state and conditions, while at the same time ensuring more privacy since neither the sensor data, nor the actual high-level context ever leave the phone. On the other hand, classifying the data on the server requires significantly less time and thus allows the usage of much larger data models, which would otherwise be unfeasible on the phone due to the big classification times. Furthermore, it is trivial for PCD-S to adapt the data model when new data or feedback comes from the user regarding already classified data and it would require no actions from the user, whereas PCD-P would require all users to install an upgrade or download the new model. Finally,

PCD-S causes less impact on the phone battery, since all the energy-expensive data processing and context computations are done on the server.

For both approaches, a mobile application listens for events associated with incoming phonecalls, notification, and alarms. When such an event is received, the application records a fragment of the ringtone together with the environment sounds and sends it to the next step. For PCD-P, next steps are executed by the same application, while for PCD-S passing the recording to the next step in the pipeline involves transferring it to a server.

Our processing pipeline is very similar to the one used in the previous approach. Therefore, we will only highlight particular aspects and differences, without an in-depth description of each stage. The five steps are conceptually the same for both architectures, differences lying only in the implementation. First we apply a rectangular windowing function with a fixed length window size to the recordings. Then the samples undergo silence removal. This step is important since ringtones and alarm sounds are repeated multiple times, with breaks in between. Thus, depending on the length of the recording, it might contain windows that are made up of environment sounds or completely silent windows. The silent windows will be removed, however the windows containing only environment sounds cannot be removed in this step. However, when having more than one window to classify, the classification result is obtained through a voting system; thus, the windows containing the ringtone will outnumber and most likely overcome the impact of windows containing only environment sounds. Furthermore, there are additional challenges in the case of very short audio tracks, which are thus repeated more times during a recording. In this case, more data is lost and thus same length recordings of different ringtones lead to different classification accuracies.

The processing pipeline is similar to the one used for the previous approach. First, we apply a Fourier transform to convert the signal from the time domain to the frequency domain. Afterwards, in order to reduce the dimension of the data, we extract the feature array. Similarly to the previous approach, we use MFCC and Delta MFCC as main features. Given the similarity of the two audio classification tasks, we expect the same features to provide good results. In the last step, the feature arrays are classified using one of the following classifiers: Decision Trees (DT), Random Forest (RF), Gaussian Naïve Bayes (GNB), K-Nearest Neighbors (KNN), and Gaussian Mixture Model (GMM).

5.3.2.2 *Evaluation*

In this section we evaluate our system from multiple points of view. We first describe our evaluation setup and present an overview of the types of recordings we use. We then proceed to investigate feasibility issues for PCD-P and PCD-S and to compare the energy and time efficiency of the two models. Afterwards we evaluate and compare the results of the different types of features and classifiers in determining phone position based on ringtones, alarms and notification sounds respectively.

For the evaluation, we used three different smartphone models: Samsung Galaxy Nexus, Samsung Galaxy S3, and LG Nexus 5. All three models can support the 44.1 kHz sampling rate and the 16 bit audio depth required for an optimal classification.

First of all, it is essential to clarify the feasibility issues of PCD-P and PCD-P with regards to processing times and battery consumption. Additionally, for PCD-S we investigate the impact of sample upload times on the classification and thus phone mode adaptation time.

Since PCD-S and PCD-P carry out the sample collection step in an identical fashion, the energy consumption for this step is the same. The two approaches differ only in the data processing step, one app sending the files to the server to be classified and the other app classifying them locally. Therefore, we compare the energy consumption of the data processing step. For this purpose, we measured the drop in battery percentage caused by processing one hour's worth of samples for the two cases. This means, for a 10-fold cross validation approach, that the system was trained with nine hours' worth of recordings and evaluated on one hour of recordings. The amount of test data is considerably more than what one could expect for an actual user even over a whole week. Considering the window size of about 100 ms (92 ms to be specific), one hour of recording would correspond to more than 36000 phonecalls, notifications, and alarms. Thus, PCD-S has lead to a 4% drop in battery percentage, while PCD-P has caused a 36% drop in battery percentage.

Classification techniques used by PCD-P and PCD-S are the same and therefore yield the same results. In what follows we analyze the performance of the various types of features and classifiers for the ringtones, notification messages, and alarm sounds without further discussion of the architecture used.

In order to evaluate the impact of ringtones as pilot sequences, we used all pre-installed ringtones from Samsung Galaxy Nexus and Galaxy S3 phones. Samsung Galaxy Nexus has 25 pre-installed ringtones, with an average duration of 8.5 seconds, while Galaxy S3 has 34 pre-installed ringtones, with an average duration of 25.2 seconds. With very few exceptions, the original audio files that came with the phone were different for each model. In addition to classifying the four considered phone positions, we also distinguish between the two possible ways of placing the phone: facing the ceiling or facing the desk. We used 5,474 samples, totaling more than 880 minutes, collected in both silent and noisy environments (home, office, outdoors, public transportation means, shops, restaurants, cafés). All samples were labeled by the users collecting them and for the evaluation we used a 10-fold cross-validation approach.

Figures 24 and 25 present the classification results for the Samsung Galaxy Nexus and Samsung Galaxy S3 ringtones respectively. We highlight on the plot two particular results. One of them corresponds to the ringtone "Over the Horizon", one of the most popular Samsung ringtones, the results of which we discuss in more details later on. The other highlighted result is obtained when recording environment sounds alone. It is obvious from both figures that using ringtones as pilot sequences has a significant positive effect on classification results. Thus, for Samsung Galaxy Nexus precision and recall increase from 77.07% and 77.07% respectively, to an average precision and recall over all ringtones of 94.37% and 94.37%. When using a Samsung S3, precision and recall increase from 84.06% and 84.06% when using no pilot sequence to an average of 95.57% and 95.54%. One aspect to be noted is that even the performance when probing passively varies between the two phone models, due to the differences in the microphones used. An additional reason can be that Samsung Galaxy Nexus has a more aggressive noise cancelling mechanism and thus reduces the negative impact of noise on classification results.

One can notice there is a bit of variation between the results obtained for the various ringtones. Thus, for Samsung Galaxy Nexus precision and recall vary between 98.98% for "Aquila" and "Argo Navis" and 90.03% for "Zeta". These differences stem on the one hand from the particularities of each signal and on the other hand from the training and classification process itself. Ringtones consisting of shorter audio snippets/clips are repeated more often and thus recordings used for training as well as those used for classification contain more windows with just environment sounds, and we have already shown that environment sound classification yields worse results overall. This issue affects both model training and evaluation itself, results being slightly worse. It should be noted however that even the worst-performing ringtones still yield significantly better results than recordings of environment sounds alone.The training issue could be solved by manually removing the breaks between ringtone repetitions in the training set. We cannot however guarantee that recordings of environment sounds alone are not used for the evaluation.

In a real-life deployment recording length would correspond to the window size, i.e. 92 ms, so it would be impossible to reach the end of a ringtone, however there is a possibility that the recording starts before the ringtone being played due to the operating system's event scheduling. Thus, since there is a chance that a test sample contains no pilot sequence, we chose to keep such samples in the training set as well.

As previously mentioned, Figures 24 and 25 highlight the results corresponding to "Over the Horizon", one of the most popular Samsung ringtones. In addition to the previous phone models, we evaluated the approach on LG Nexus 5 phones as well. In what follows, we look closer into the results of this ringtone, focusing our discussion on the Samsung Galaxy Nexus to avoid unnecessarily complicating the discussion. Overall findings are similar on Galaxy S3 and LG Nexus 5, with the small variations in numbers inherent to the difference in hardware and operating system.

Figure 26 shows the precision and recall of the different combinations of feature arrays and classifiers that we experimented with. As far as types of features are concerned, MFCC and Delta MFCC stand out for achieving the best results regardless of the overall performance of the classifier and its suitability for the current problem. Decision Trees and Random Forest are less affected by the type of feature than other classifiers, having good and very good results respectively regardless of the feature vectors used.

Thus, Random Forest yields the best results out of all classifiers we considered. In contrast to this, the second best classifier, K-Nearest Neighbors is very sensitive to the feature array that is used. Thus, it offers very good results for MFCC and Delta MFCC, but is quite inaccurate when using band energy. Similarly to the previous approach, one can notice that the data is too complex for Decision Trees, which cannot model it as well as Random Forest or k-Nearest Neighbors.

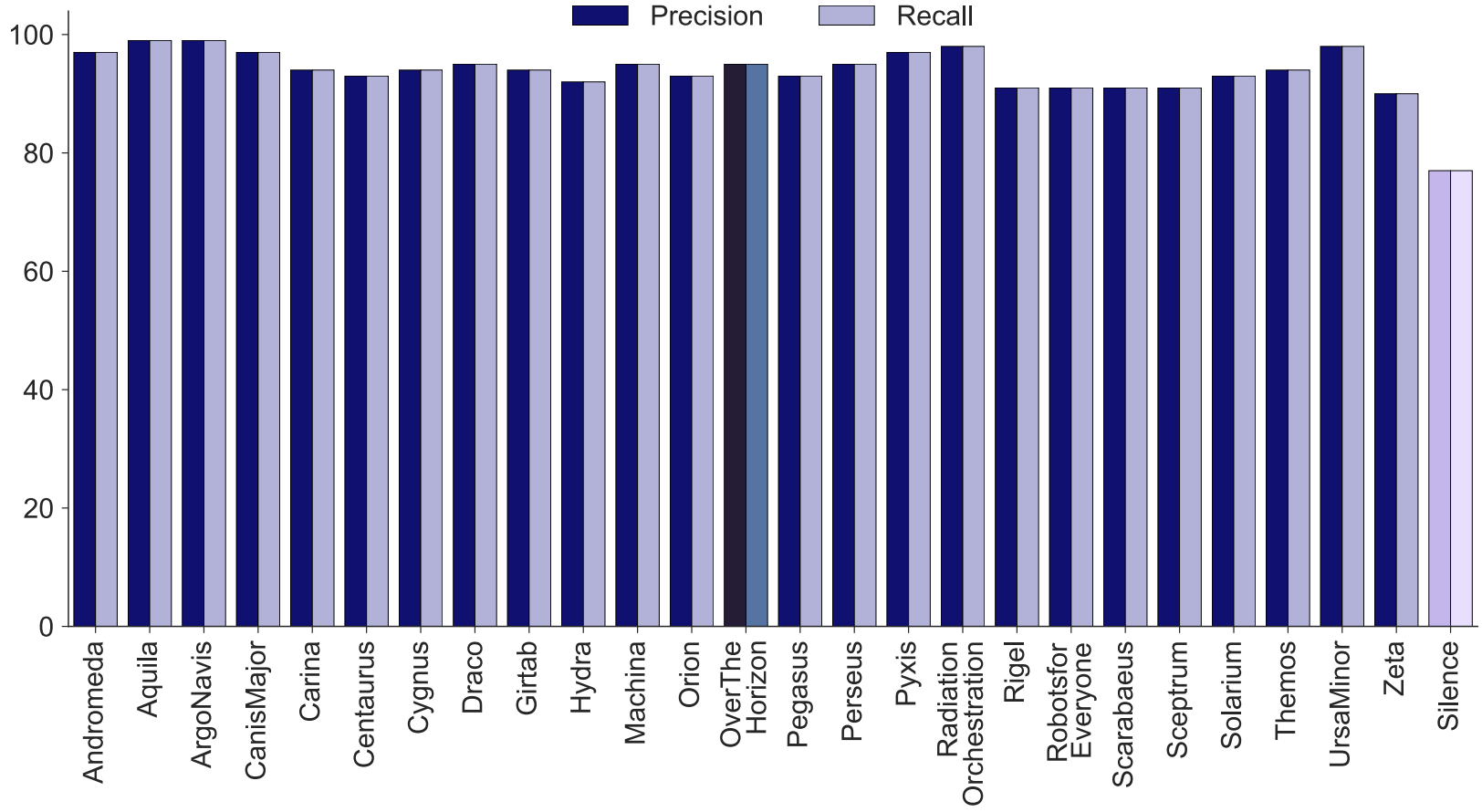


Figure 24: Classification results for all Samsung Galaxy Nexus ringtones when using MFCC and Random Forest

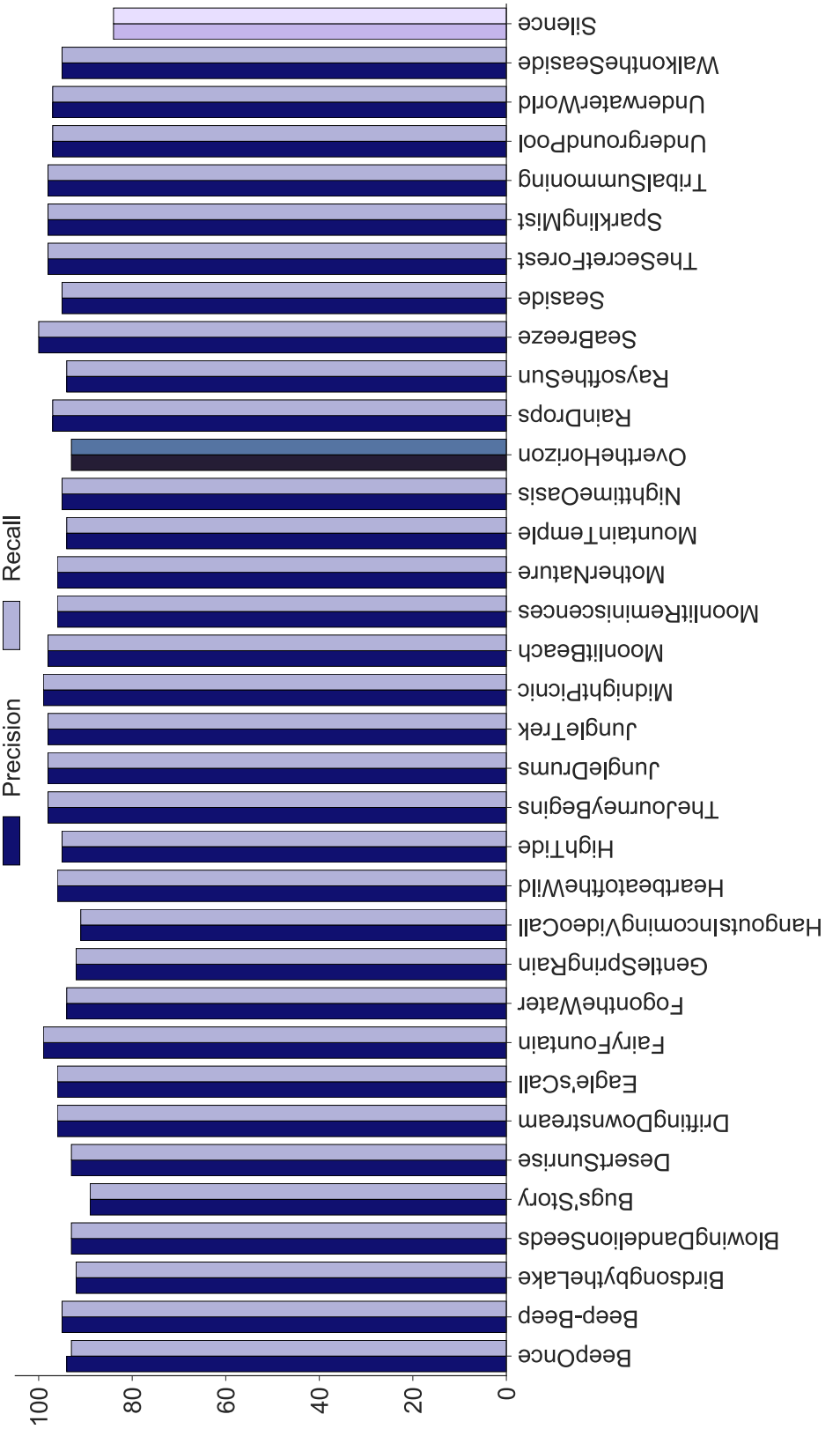


Figure 25: Classification results for all Samsung Galaxy S3 ringtones using Delta MFCC and Random Forest

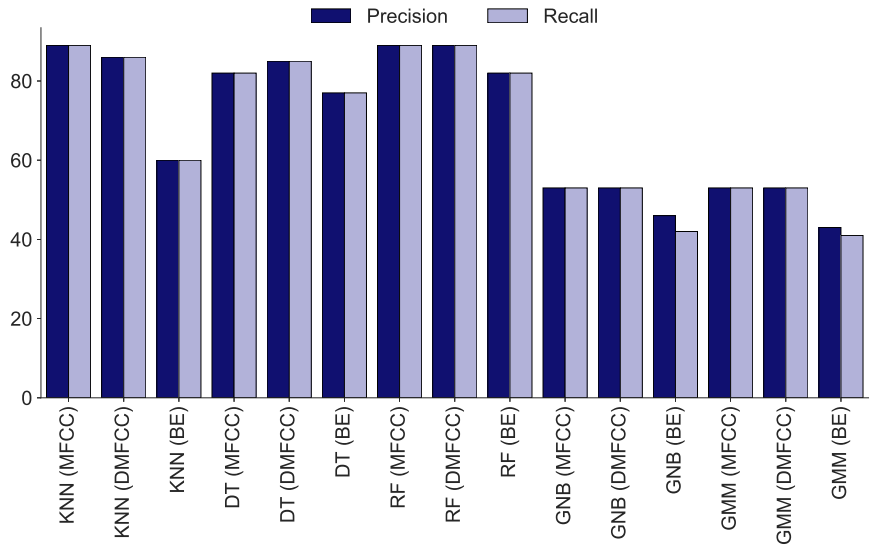


Figure 26: Comparison of classifier results for "Over the Horizon"

Next, we look closer at the classification results of Random Forest and MFCC, which obtained the best results. It is unsurprising that Delta MFCC yielded slightly worse results that MFCC. Delta MFCC is more suited for situations where the temporal aspect plays a significant role, for instance when trying to detect user activity, or phone environment at the same time with phone position. The impact of phone position itself on the recorded signal is not affected by time and a single window is sufficient for the classification. Therefore, taking into account the temporal aspect like Delta MFCC does, and thus doubling the size of the feature vector is unnecessary and even causes slightly worse results since the classifiers have to deal with higher-dimensional data. Figure 27 presents the confusion matrix for the phone position classification using the afore-mentioned features and classifier, while classifying also the environment noise level. The confusion matrix shows there are actually quite few misclassifications, most of them happening in noisy situations. Again, a significant

True class \ Predicted class	bagn	bags	desn	dess	hann	hans	pocn	pocs
bag_noisy	87.2	0.6	0.3	0.4	2.6	0.6	3.9	4.4
bag_silent	0.5	89.1	0.0	0.3	4.1	0.5	3.1	2.5
desk_noisy	3.8	0.6	79.4	0.5	3.1	11.1	1.0	0.6
desk_silent	1.7	0.4	1.0	91.6	0.7	3.0	1.1	0.7
hand_noisy	0.8	1.5	0.6	0.1	85.6	4.0	5.6	1.8
hand_silent	1.2	1.1	4.3	1.4	7.2	83.8	1.0	0.2
pocket_noisy	1.8	0.8	0.3	0.6	9.8	0.8	81.5	4.5
pocket_silent	2.3	1.1	0.3	0.5	2.7	0.7	3.5	89.1

Figure 27: Classification results (in %) for position and environment noise level when using "Over the Horizon" as pilot sequence, based on 31394 windows

numbers of misclassifications occur between similar phone position, i.e. the phone being either inside or outside of an enclosure.

In what follows, we focus on the impact of using alarm sounds on the phone position classification results. Thus, we look into the results of the six Samsung Galaxy Nexus alarm sounds. Again, Random Forest was the classifier that yielded the best results on average. Figure 28 compares the F-score results of the classification for all alarm sounds when using two different types of features, MFCC and Delta MFCC.

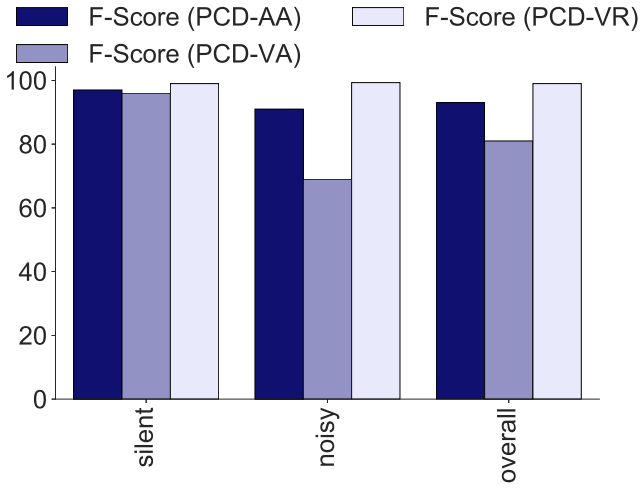


Figure 28: Phone position classification results for all Samsung Galaxy Nexus alarm sounds

One can notice that "Cesium" and "Hassium" have slightly worse results than the other alarms. This is caused by the shorter duration of these audio tracks, which leads to 50% or more of the recordings to contain only environment sounds. This affected both the training of the model, as well as the classification process itself. However, even the results of these two alarm sounds are excellent per se. "Cessium", the default Samsung alarm sound, had a 96% F-score when using Delta MFCC and Random Forest. As shown in Figure 29, most confusions between the phone positions are negligible, the only one standing out being the one between the desk and the user's hand, so positions which are closer to each other, the phone being outside of an enclosure in both cases.

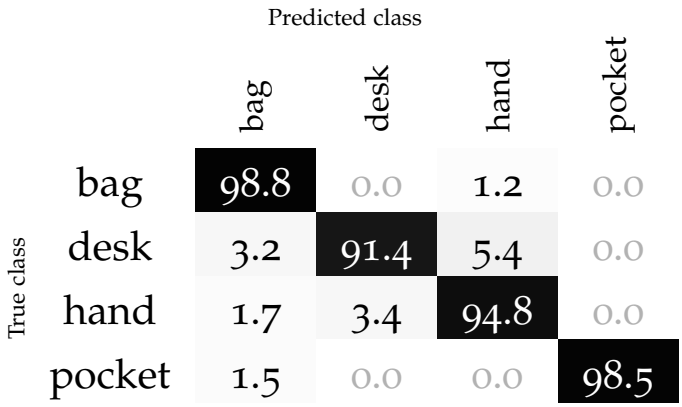


Figure 29: Classification results (in %) for "Cesium" alarm sound, based on 300 windows

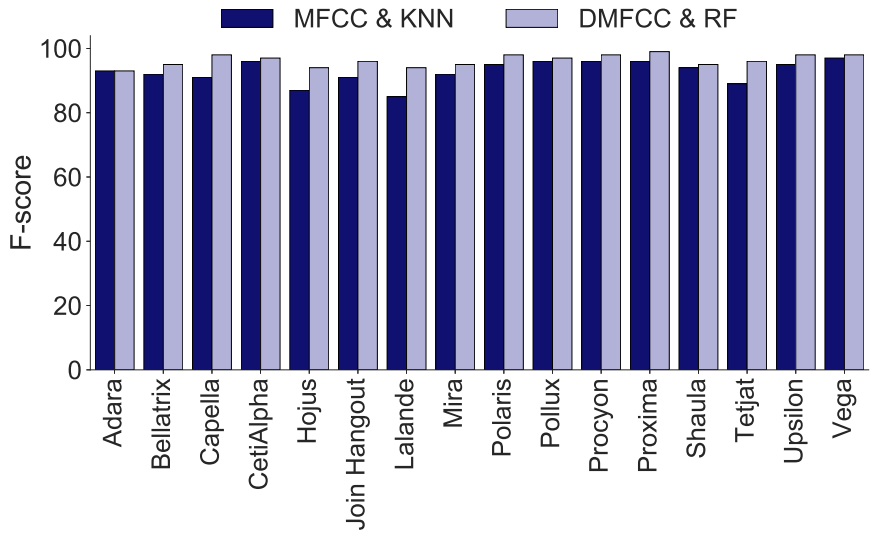


Figure 30: Phone position classification for all Samsung Galaxy Nexus notification sounds

Next we discuss the results obtained when using notification sounds as pilot sequences. We evaluated our solution on the 17 notification sounds that are included by default in the Samsung Galaxy Nexus phones, gathering over 2,000 sound samples. The clips have lengths between 0.6 and 4.4 seconds and are made up of up to 58% silence.

Figure 30 compares the F-scores yielded by all the different notification sounds when classifying phone position. We used the two combinations of features and classifiers that provided the best results, K-Neareast Neighbours with MFCC and Random Forest with Delta MFCC. One can notice again that the sound files that contain significantly longer periods of silence percentage-wise, like "Hojus" and "Lalande" have also lower F-scores.

5.3.2.3 *Achievements and Remaining Challenges*

To sum up, the current approach succeeded in using opportunistically gathered ringtones, alarms, and notification sounds as pilot sequence. As proven above, we can determine phone position with very good results regardless of the audio signal captured.

There is an obvious variability of results, precision and recall varying on Samsung Galaxy Nexus between 98.98% for "Aquila" and "Argo Navis" and 90.03% for "Zeta". The average precision and recall are 94.37% for Galaxy Nexus, which is an improvement of 22.44% to using recordings of environment sounds alone. This represents only a slight decrease from the previous approach's best result which was obtained when using Gaussian noise as pilot sequence, with a precision and recall of 98.76% and 98.91% respectively.

We deem this to be a profitable tradeoff, given the guarantee of having the user's current phone position, and not only a recent one. Additionally to this, we have also energy savings and reduced privacy concerns stemming from the avoidance of duty cycling.

This approach also reduces the disturbance risk, but cannot completely eliminate it, since the few hundred milliseconds will be audible in silent environments. A so-

lution to this issue would be a combination of duty cycling and the opportunistic approach, together with a bias of the phone position-phone mode mapping towards silent phone modes. An additional issue of the current approach is that it is applicable only if the phone is in a normal or loud mode, since it relies on sound emissions. Therefore, the next subsection is dedicated to solutions for the phone's silent mode, whereas solutions for the "only lights" more are presented in Section 5.4 which concerns sensor fusion approaches. In the latter case though, opportunistic active probing approaches are not feasible.

5.3.3 *Using Vibration Motor-triggered Movements as Pilot Sequences*

For the case when the phone is on silent mode we explored another possibility: using the movements triggered by the vibration motor as pilot sequences. There are additional advantages of this method. First of all, it entails less privacy concerns since no audio recordings are gathered and involves no risk of user disturbance. Furthermore, classifying the phone position based on the accelerometer data consumes less energy and is not affected by environment noise. However, there are also drawbacks when using accelerometer compared audio traces. Accelerometers have significantly lower and more variable sampling rates compared to microphone, which requires longer recording times to ensure sufficient readings are provided for phone position classification.

5.3.3.1 *Concept*

The overall idea behind this approach is similar to the previous one. Signals will propagate differently in the various phone positions, thus creating different fingerprints for these positions. This stands true regardless of whether that signal is sound or movement. For example, it can be muffled in an enclosed space, like a bag or a pocket, or amplified on a hard surface. Furthermore, the movements triggered by the vibration motor would be impacted by phone position and have a more pronounced fingerprint than just phone movement alone.

In order to be able to compare approaches, we shall name them as follows: PCD-AA (Phone Context Detection – Audio & Audio), PCD-VR (Phone Context Detection – Vibration & Accelerometer Readings), and PCD-VA (Phone Context Detection – Vibration & Audio). PCD-AA represents the approach presented in the previous subsection, which classifies phone position based on audio recordings, while opportunistically using ringtones and phone notification sounds as pilot sequences. PCD-VR is the main application we shall present in what follows, which classifies accelerometer readings. Finally, PCD-VA investigates whether sounds produced the vibration motor being triggered can be used as pilot sequences for audio recordings and improve classification results or at least obtain similar results to PCD-AA.

In what follows we shall focus on presenting PCD-VR. PCD-VA is very similar to PCD-AA, using the same classification pipeline and same implementation, with the only difference that instead of ringtones and notification sounds, vibration motor triggered sounds are recorded. We shall discuss and compare the three approach in the Section 5.3.3.3.

As far as the system architecture for PCD-VR is concerned, we have considered, similarly to the previous approach, both the case when all processing is done on the

phone, as well as the case when the processing is done on a server. The processing pipeline is shown in Figure 31. When a phonecall, notification, or alarm comes in, our app collects accelerometer samples. One issue we encountered was that the accelerometer sampling rates are much lower than the microphone's, as well as more variable. However, one needs also a very limited number of samples for the classification. Thus, we experimentally determined that eight samples constitute an optimal window length.

As far as the system architecture for PCD-VR is concerned, we have considered, similarly to the previous approach, both the case when all processing is done on the phone, as well as the case when the processing is done on a server. The processing pipeline is shown in Figure 31. When a phonecall, notification, or alarm comes in, our app collects accelerometer samples. One issue we encountered was that the accelerometer sampling rates are much lower than the microphone's, as well as more variable. However, one needs also a very limited number of samples for the classification. Thus, we experimentally determined that eight samples constitute an optimal window length.

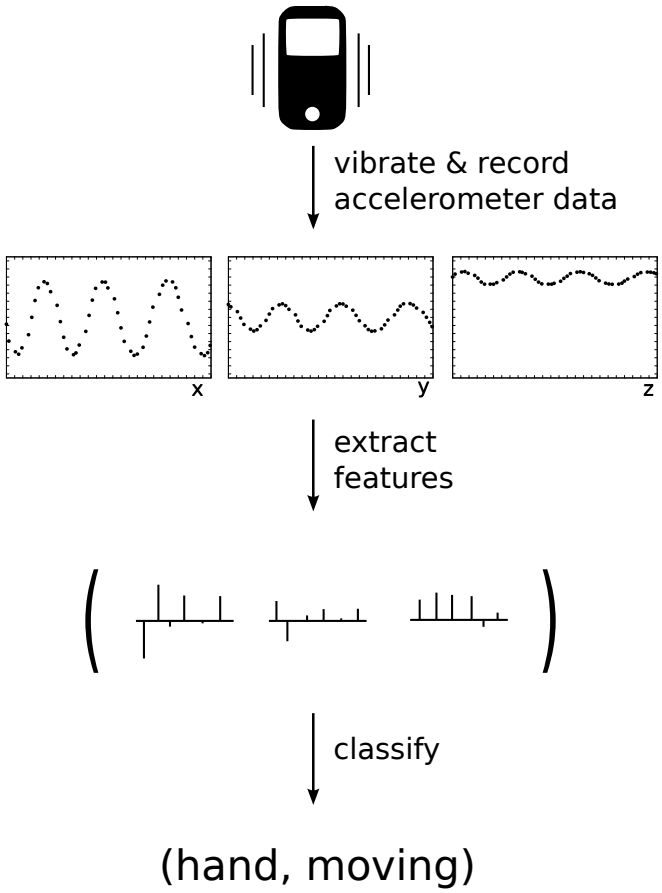


Figure 31: Processing pipeline for PCD-VR

We did not remove the samples collected when the phone was not vibrating, since our approach classified only one 110 ms window, in order to allow for the phone mode adaption to be made very promptly. Even for the hybrid approaches we will describe in the following section we will be recording at most 10 windows, which would still bring us nowhere near the end of the 1 500 ms of vibrations.

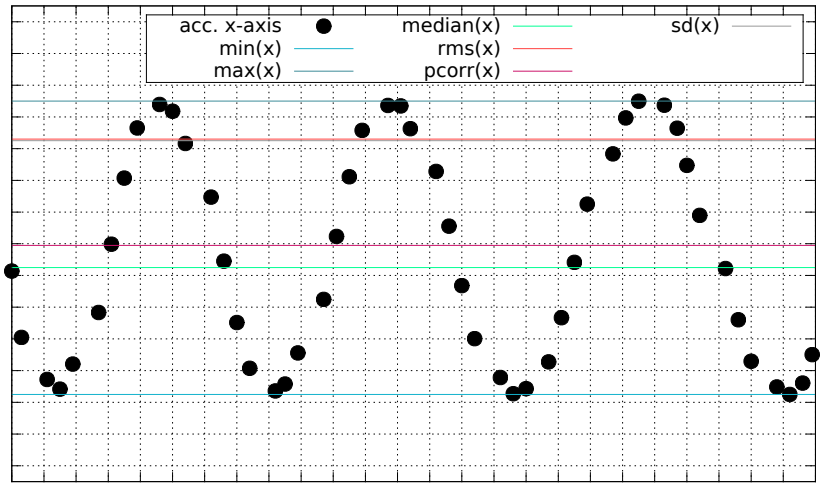


Figure 32: Features for the x-axis component of one acceleration window

After collecting the samples, we extracted the feature vector for each window, using a combination of features that have been successfully used in various combinations to classify movement [2, 5]. The structure of our feature vector is presented in Equation 15.

$$\begin{aligned} \textit{features(data)} = (&\min_{x,y,z}(\mathtt{data}), \max_{x,y,z}(\mathtt{data}), \mathtt{median}_{x,y,z}(\mathtt{data}), \\ &\mathtt{rms}_{x,y,z}(\mathtt{data}), \mathtt{pcorrelation}_{x,y,z}(\mathtt{data}), \mathtt{sd}_{x,y,z}(\mathtt{data})) \end{aligned} \tag{15}$$

Where $\min_{x,y,z}(\mathtt{data})$, $\max_{x,y,z}(\mathtt{data})$, and $\min_{x,y,z}(\mathtt{data})$ are the minimum, maximum, and median value of the x, y, and z components of the acceleration data gathered for one window. $\mathtt{rms}_{x,y,z}(\mathtt{data})$, $\mathtt{pcorrelation}_{x,y,z}(\mathtt{data})$, and $\mathtt{sd}_{x,y,z}(\mathtt{data})$ are the root mean square, Pearson correlation, and standard deviation of the data along the three axes. This gives us a total of 18 coefficients for our feature vector. Figure 32 shows 6 of these coefficients, extracted for the x-axis component of one acceleration window.

The last step is the classification. Here we used the same classifiers as for the previous approaches, since we are looking for similar patterns in the data. Additionally, the classification tasks have even similar dimensionality: in the audio case we consider feature vectors with 13 or 26 elements, while in this case feature vectors have 18 elements. Thus, the machine learning algorithms we considered were Decision Trees (DT), Random Forest (RF), K-Nearest Neighbors (KNN), and Gaussian Mixture Model (GMM).

5.3.3.2 *Evaluation of PCD-VR*

For evaluating this approach we used three phone models, Samsung Galaxy Nexus, Samsung Galaxy S3, and LG Nexus 5, with which we have collected over 1 300 000 accelerometer samples. We gathered the samples in a variety of situations: silent and noisy, stationary and moving alike. While noise level does not influence accelerometer readings, we have collected the data in settings similar to those of the previous approaches in order to be able to compare the results. We take into account and even classify the user's mobility status, i.e. whether he is moving or stationary since this

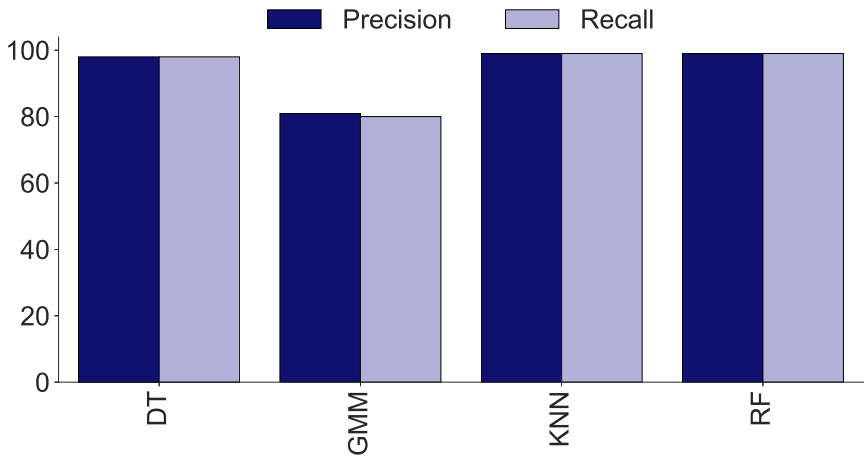


Figure 33: Comparison of overall precision and recall for all classifiers

is one factor we expect to influence classification results. For the evaluation we use a 10-fold cross validation approach.

We will first focus on the results obtained when classifying phone position alone, and not on the user's mobility status (whether he is moving or not). Figure 33 offers an overview of the performances of the different classifiers we experimented with. Similarly to the previous approaches, Random Forest and K-Nearest Neighbors are the classifiers that yield the best results. They are followed closely by Decision Trees, which is an important finding we can leverage in a real-world deployment of our system. The low complexity and computation time make Random Forest particularly appropriate for the phone mode adaptation requirements, as we discuss in more detail in Chapter 6. Gaussian Mixture Models perform significantly worse, with a precision of 81.5%, compared to the 99.2% achieved by Random Forest.

These results are similar to our audio-based approaches, where Random Forest and K-Nearest Neighbors were the best performing classifier. This is not surprising since phone position will have analogous effects on the different sensor readings. In a pocket, both sound and phone vibrations will be muffled.

Figure 34 shows in more details the results provided by the best performing classifier, namely Random Forest. The overall precision and recall are 99.20% and 99.08% respectively. As the plot shows, the results were almost perfect, except that for less than 1% of the situations the user's hand was classified as a bag.

In the second step, we classify both the phone position and user movement. We consider all combinations of phone positions and user movement, except for the case when the phone is on a desk, since a desk in motion is a quite unusual occurence. For the classification we obtain an outstanding precision of 99.23% and a recall of 99.21%. It is worth noting that we added a whole new dimension to our data by determining whether the user is stationary or moving, which caused a drop in precision of only 1.4%. Figure 35 shows the corresponding confusion matrix, when using Random Forest as classifier. The extra misclassifications did not affect the quality of classification of phone position, i.e. in most cases the phone position is still correctly classified while the confusion occurs between the stationary and in motion state of the user. Most such confusions occur for phone positions which naturally involve a higher amount of movement, even if the user is stationary per se. Thus, in the de-

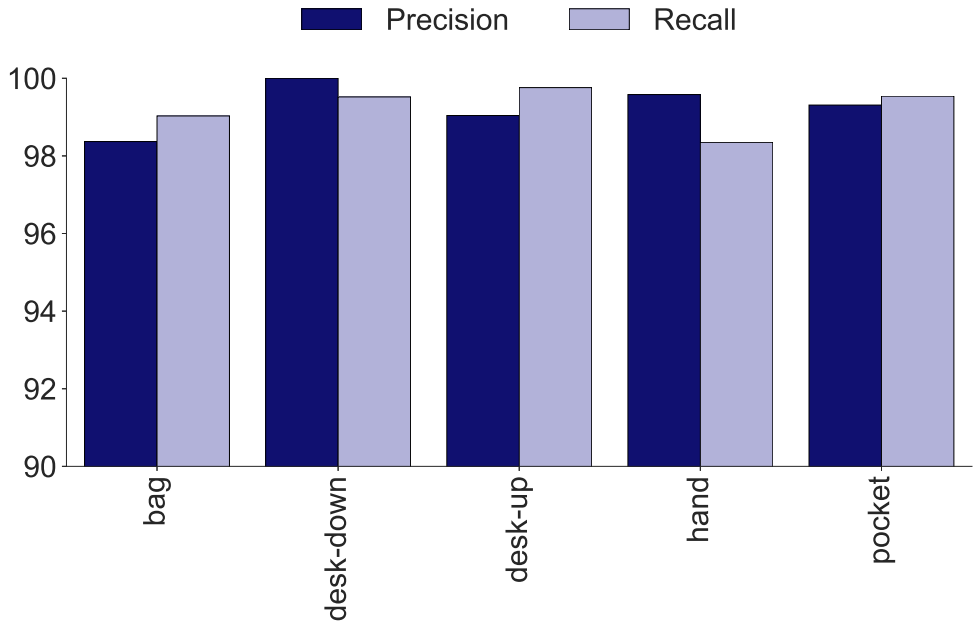


Figure 34: Classification precision and recall scores for Random Forest

creasing order of the number of confusions, these positions are pocket, hand, and bag.

As a conclusion, using the vibration motor as pilot sequence for an accelerometer readings-based classification approach leads to outstanding results, with precision and recall in excess of 99% when using Random Forest as classifier. The findings are particularly relevant since the accelerometer readings are not affected by environment noise and this approach can still be used when the phone is on silent mode. Additionally, while about 100 ms of an audio snippet might still be perceived in very silent environments, the phone vibrations are very unlikely to be noticed or

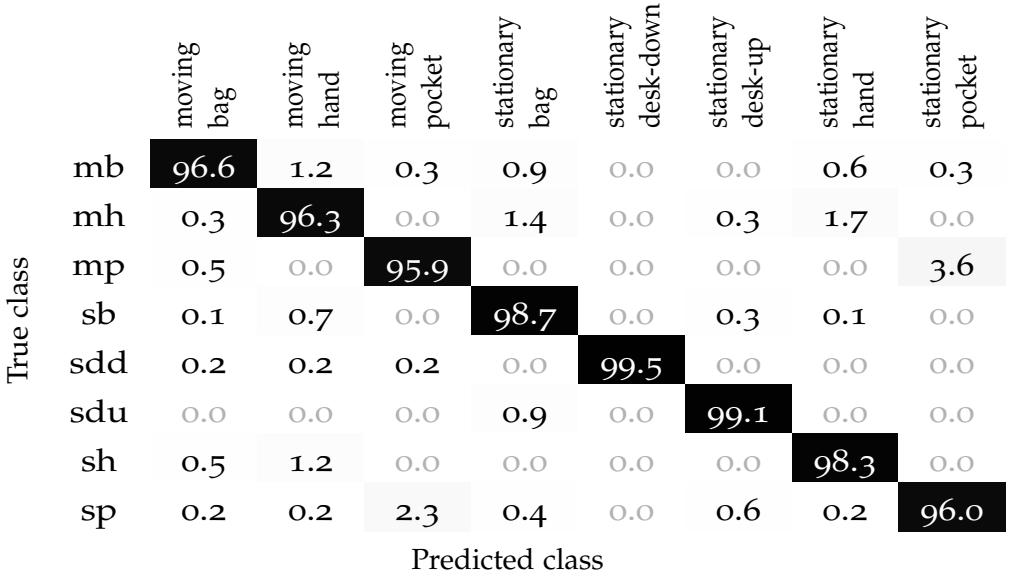


Figure 35: Confusion matrix for phone position and user movement classification using Random Forest (in %)

cause any disturbance. In the following section we compare the results yielded by the audio and accelerometer-based approaches, i.e PCD-AA and PCD-VR, as well as PCD-VA.

5.3.3.3 *Comparison of Audio and Accelerometer-Based Active Probing Approaches*

As above-mentioned results have shown, active probing methods that rely on audio recordings and accelerometer readings provide excellent classification results for phone position, results which significantly exceed those obtained just by passive probing. Furthermore, the microphone and accelerometer are the only sensors that can be used to distinguish between all phone positions we considered, and not just classify whether the phone is inside or outside of an enclosure. In what follows, we will compare the performance and individual advantages of each approach.

Additionally to the two afore-presented approaches, we experimented with recording audio traces of the triggered vibration motor in the various phone positions. The whole processing and classification process followed the same steps as the approach for piggybacking notification sounds. We carried out this experiment in order to investigate whether audio recordings could still provide good enough results even if the pilot sequence was not a ringtone or notification sound but rather the sounds produced by the vibration motor.

Figures 36 presents a comparison between overall precision and recall obtained by the best performing features and classifiers for three approaches: 1) PCD-AA, which uses incoming ringtones as pilot sequences; 2) PCD-VA, which uses the sounds triggered by the vibration motor as pilot sequences, or in other words; 3) PCD-VA which uses the movements triggered by the phone's vibration motor as pilot sequence for accelerometer readings. One of our goals is to compare the performance of the three methods in sub-optimal situations for the two sensors, i.e., noisy situations when using the microphone, and a user/environment in motion for the accelerometer, and thus experimentally establish the fitness of each method for the corresponding situations. However, as already shown in the previous section, impact of movement on accelerometer readings classification results is negligible (1.4%) and therefore we only compare the results of the three methods in silent, noisy only, and a combination of environments.

As expected, PCD-AA and PCD-VA perform best in silent situations. PCD-AA is only slightly affected by noise, whereas the impact of noise on PCD-VA is quite significant. This is more obvious in noisy situations alone compared to the combination of situations. PCD-VA is more affected by noise than PCD-VA due to the low volume of the vibration motor sounds, which get more easily covered by environment sounds. On the other hand, PCD-VR achieves noticeably better results in all situations, particularly in the noisy and combined situations. Thus, its F-score in noisy environments is 9.8% better than that of PCD-AA and 43.4% better than that of PCD-VA. In combined situations, it obtains 6.6% and 23.5% better F-scores than PCD-AA and PCD-VA respectively. Obviously, accelerometer readings are not affected by acoustic noise. This, together with the lower energy consumption or network traffic, as well as the practically zero potential for user disturbance make this method the most fit one for real life deployments.

As a conclusion, active probing can achieve excellent results by using either audio or accelerometer readings. Accelerometers provide slightly better results and do not

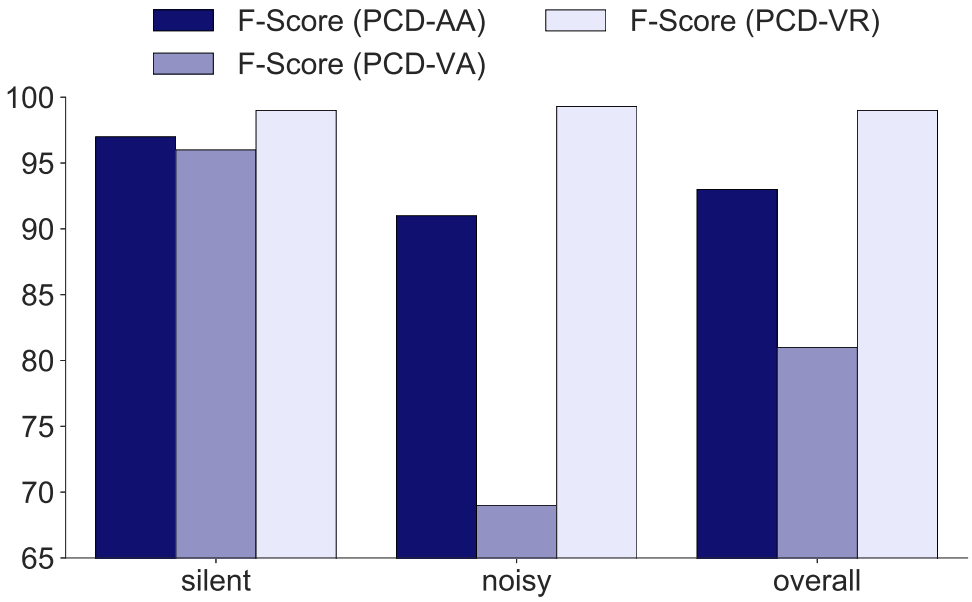


Figure 36: Comparison of overall F-scores for PCD-AA, PCD-VA, and PCD-VR in silent, noisy, and combined situations

suffer from environment noise. At the same time, privacy concerns associated to collecting accelerometer data are very much reduced in comparison to recording audio samples, regardless of the recording duration. However, accelerometers have much lower sampling rates and thus need longer recording times on most phone models. Additionally, accelerometers have significantly higher impact on battery consumption.

5.3.3.4 *Contributions and Remaining Challenges*

To sum up, the current approach determines phone position with a precision and recall of up to 99.23% and 99.21%, bringing thus a improvement to the ringtone-based approach of up to 10% depending on the ringtone used. It is important to note that this classification performance is no longer influenced by a user decision, namely the choice of ringtone. The usage of accelerometer instead of microphone traces involves less privacy concerns, while at the same time avoiding disturbance risks, since a few hundred milliseconds of phone vibrations are practically inaudible. Furthermore, accelerometer samples classification is not influenced by ambient noise. A potential concern was represented by user or environment movements, however their impact was shown to be negligible by the evaluation presented in Section 5.3.3.2. The only issue that stays open even with this approach is classifying phone position when the phone is on "only lights" mode. In that case, active probing is not possible and for this reason we have to rely only on combinations of various sensor traces, as we show in the following section.

5.4 SENSOR FUSION

As previously shown, audio and accelerometer can be used with great success to classify phone position provided that we not only passively probe the environment

but we also use pilot sequences. With results in the range of 98-99% precision and recall, there is little improvement left in order to justify usage of additional sensors. Furthermore, the only other sensors that we could use are the light sensor and proximity sensor, which can only distinguish whether the phone is inside or outside of an enclosure. Such misclassifications between, for instance bag and desk or pocket and hand are extremely small, as shown by Figure 35, and therefore do not justify the extra issues and drawbacks brought by using additional sensing modalities.

There is however one situation when active probing is not applicable: when the phone is on "only lights" mode. In this case, there will be no ringtone played and the vibration motor won't be triggered. Therefore, the classification can rely only on the selection and combination of sensing modalities in order to improve the classification performance. It should be noted that we want to keep using an opportunistic approach, i.e., collect sensor samples when a phonecall or notification comes in. While this does not provide us with free pilot sequences, it still allows us to avoid duty cycling and guarantees the currentness of the classified phone position.

5.4.1 *Concept*

Recordings of environment sounds yield a precision and recall between 77 and 84%, depending on the phone model, while passive probing accelerometer readings reach a precision and recall of about 75% [38, 158]. Therefore it might be a sensible decision to combine the two sensing modalities. This is made possible by the Nexus 5 accelerometer sampling rate which provides us with sufficient samples in 110 ms, which is still an acceptable window size for audio, compared to the regular 92-100 ms window size.

As discussed in Section 5.2, we could additionally use the light and proximity sensor to improve classification results. Both sensors however can just determine whether the phone is within or outside of an enclosure. Therefore, they can be used to provide additional information when the audio and accelerometer classification results do not coincide and, furthermore, they disagree whether the phone is "inside" (pocket, bag) or "outside" (hand, desk).

Table 5 presents the times required to get a sufficient number of samples for the light and proximity sensor, together with the number of samples obtained in 100 ms, which is the recording length for audio and accelerometer settings. These values were obtained experimentally for sensors built in Nexus 5 phones, since the values provided by the manufacturer referred to the maximum achievable frequency, which was infeasible for situation when multiple sensors are simultaneously sampled; additionally, multiple applications running on the phone, which is the realistic case for regular users, further affect this rate. For the light sensor we aimed to obtain eight readings, as shown by preliminary tests to be sufficient, and for the proximity sensor a single reading. Due to the recording times, the users would most likely hear the notification sounds. Therefore, both sensors are not suited for the opportunistic approach.

Therefore, our approach would be to concomitantly collect audio and accelerometer samples and to classify them following the steps presented in the previous sections. As a method to boost classification accuracy, we propose to collect more than just one window for each sensor. This is anyway unproblematic since there is no dis-

Sensor	Minimum recording length to get sufficient samples	Samples obtained in 100 ms
Light sensor	400	2.6
Proximity sensor	dependent upon change in lighting	0

Table 5: Experimental results of light and proximity sensors frequencies on Nexus 5 phones

turbance source for the user, the incoming phonecall or notification being signaled just by the screen lighting up. Using this approach we can use a voting system to decide on the current phone position and thus also make the decision easier in case audio and accelerometer readings classification results differ.

5.5 FINAL CONSIDERATIONS

In the current chapter, we have proposed different methods to classify phone position, achieving considerable improvements in comparison to related work. By using pilot sequences, we have obtained accuracies of about 98%. Additionally, we have proposed methods to obtain our pilot sequences opportunistically by piggybacking phonecall and notifications sounds. This allowed us to be certain that the phone position we have classified is not already deprecated, while at the same time reducing the battery drain and privacy concerns. The results of this approach varied based on the ringtones and notification sounds selected by the user. For most sound tracks we achieved an average precision and recall of over 90%, while for both phone models we have evaluated our approach on, the average precision and recall across all built-in ringtones was approximately 95%. Furthermore, we have presented an approach that piggybacks vibration motor movements triggered by incoming phonecalls and notifications as pilot sequence while collecting phone accelerometer readings. This approach obtained the best results of all methods proposed, while at the same time not being affected by environment noise and being suitable also for more privacy-conscious users, who would be reluctant about methods relying on audio recordings.

6

SYSTEM PROTOTYPE FOR AUTOMATIC PHONE MODE ADAPTATION

In the following chapter we present our approaches for user activity detection and subsequently explain our implementation of a phone mode adaptation application. Given that the activity detection methods per se are not novel and just the application scenarios are new, we decided to include these contributions in this chapter which concerns itself with real-life deployment issues. In the beginning we present our work on another context-aware application, where user activities are detected in order to realize an adaptive question and answering system. We then proceed to present our considerations regarding a practical implementation of phone mode adaptation and our corresponding platform.

6.1 USER ACTIVITY DETECTION FOR CAR TECHNICIANS

In what follows, we describe our approach to determine car technicians' activities based on smartphone collected audio recordings, originally published in [28]. The purpose of this approach was to support the development of a company-internal context-aware Q&A system. This system would have a similar focus on avoiding disturbances to the users as our current phone mode adaptation system. However, the approach was different. Instead of adapting the workers' phone mode, the system would instead keep track of available workers, and based on this decide to whom to forward the questions submitted in the system.

6.1.1 *Workers' Need and Willingness to Help*

Let us briefly refer to a survey that we carried out and which underlines the technicians' need for help and willingness to provide help for their colleagues when able to do so. Thus, we carried out a study on 168 car technicians. The participants consist of 36 employees from two big garages with different branches in Germany and 132 attendees in training courses offered by two training providers in Germany. Detailed results of the study can be found in [119].

One of the first concerns was to determine how often technicians needed support with their daily tasks and what kind of support they were mainly looking for. The results show a clear separation between the experts and the employees with less experience. Two thirds of the technicians report that they rarely need help, while the other third report they often need support. There is a slight increase in the number of employees requiring help when it comes to repairs carried out on the street, as opposed to the ones carried out in the garage.

When in need for help, technicians have diverging opinions about consulting the manufacturer's documentation. Thus, the respondents are split in four almost equal groups: one that rarely consults the documentation, one that sometimes consults it,

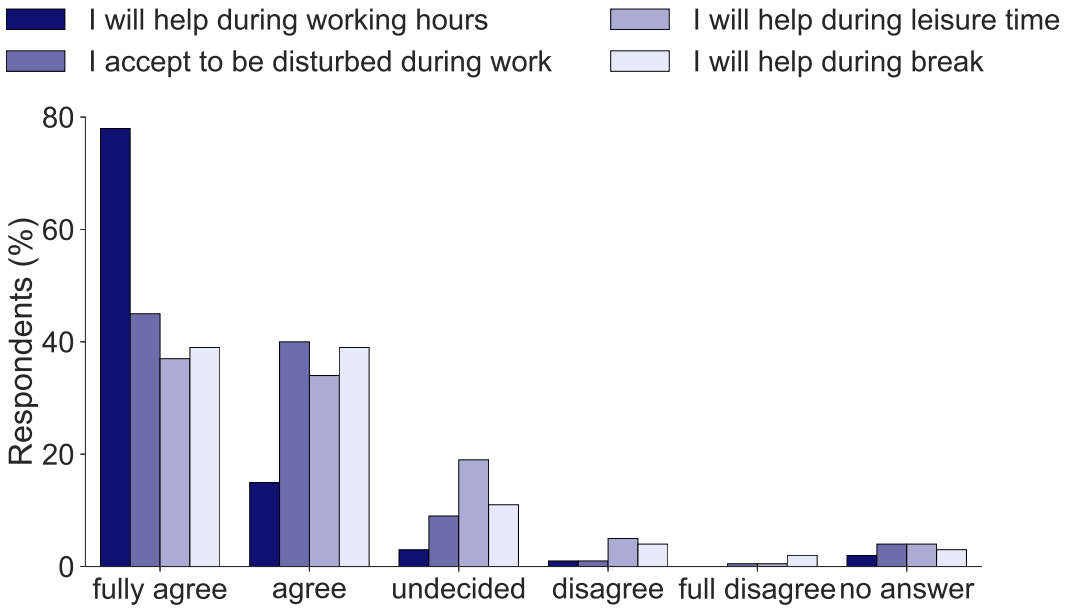


Figure 37: Comparison of the technicians' willingness to help, based on their current activity

one that often consult it, and finally one group that always consults the documentation. Asking colleagues is another popular approach, with more than half of the technicians frequently or even always relying on asking colleagues.

75% of the respondents are totally happy to help their colleagues, while 15% are in general willing to help their colleagues and 10% preferred not to answer this question.

A very important aspect here is the activity the technicians are carrying out when they are asked for help. Figure 37 shows the variation of the technicians' willingness to help over various activities. Thus, a majority of technicians readily help their colleagues during their working hours, but less then half accept to be interrupted when actually solving a task. Even a lower number of technicians are willing to use their breaks to help their colleagues. This argues strongly both for forwarding the questions during the right activity, as well as for the fact that each technician has to be able to select himself when he is interruptible or not, given the divergence of the answers on this matter.

6.1.2 *Technicians' Activities*

After determining the importance of user activity, we asked car technicians to log their daily activities for two weeks, in order to get an accurate view about their daily tasks. As expected, the most common activities were car repairments, with slightly more repairments taking place outside of the garage than within it. The next more common activities were driving by car (both for work-related tasks and commuting) and having a break. The least usual activities were meetings and other various tasks, like placing an order for new car parts. The activity set that resulted from the technicians' diary:

- Car repair in the garage

- Car repair on the street
- Other work in the garage
- Driving for work purposes
- Meeting
- Break within the garage
- Break outside the garage
- Commute to/from work
- Other activity

6.1.3 *Q&A System*

The backbone of our system is a Q&A infrastructure, that allows users to send and receive questions, as well as receive answers via multiple channels: smartphone app, web form, email, SMS. We thus comply with the technicians' mobility and different availabilities of the communications channels in their daily activities. This system is presented in detail in [119].

6.1.4 *Context Aware Notifications in a Q&A System*

As shown in the previous section, the car technicians' willingness to help their colleagues varies across their daily activities. Therefore, each technician has to decide for himself if he is available or interruptible during each type of activity. We distinguish between *availability*, the state of a user that is willing and able to help during a certain activity and *interruptibility*, the state of a user that is generally unwilling to help during his current activity, but would agree to occasionally take a break for an urgent open question.

Our goal is to detect the technicians' current activities and, based on them, their availability and interruptibility. Our centralized server would keep track of all the employees availabilities and, when a new question arrives, it would select a few available technicians to whom it would forward the question. One of the precautions would be to prefer users that received or answered less questions during that day, so that the system does not burden the same few experts while leaving all the other colleagues out of the loop.

Another reason for which we need to know the user's current activity is so that we can use an appropriate communication channel (that he has assigned himself for that particular activity). For instance, a user cannot read and type into an app while driving, and even less so use a web interface. Using voice mail to deliver the question (even taking the extra step of automated reading for text messages) and then recording his answer as a voice message would provide a significantly more appropriate solution. Table 6 shows a sample mapping between the users' activities, their availability and interruptibility, as well as the preferred communication channels.

Activity	Availability	Interruptibility	Communication channel
Car repair in the garage	false	true	app/SMS
Car repair on the street	false	false	-
Other work in the garage	true	true	app/SMS
Driving for work purposes	true	true	voicemail
Meeting	false	false	-
Break within the garage	true	true	app/SMS
Break outside the garage	false	true	app/SMS
Commute to/from work	true	true	voicemail
Other activity	true	true	app/SMS

Table 6: Mapping between activities and availability, interruptability and communication channel

6.1.5 *Activity Detection*

We consider the user activities defined in Table 6. In order to detect them we use the technicians' smartphones to record short audio samples. These are sent to our server, where they are classified following the same process described in Chapter 5. We use only passive probing, i.e. only record the environment sounds, which then undergo windowing, silence removal, Fourier transformation, feature extraction, and classification.

6.1.6 *Evaluation*

6.1.6.1 *Evaluation Setting*

To validate our approach to detect the activity, we carried out an experiment during which two car technicians used our application during their whole working time for two weeks. They both used Samsung Galaxy S4 phones, which could support a 44.1 kHz audio sampling rate. In order to collect our ground truth, the two technicians filled in separate daily forms, where they specified all their activities and their corresponding time slots.

From the all initially considered activities, we decided to leave out driving for work purposes, as this activity proved to take very little time over the two weeks of the evaluation. The situation is most likely due to the fact that most repairs that were performed outside of the garage were still in its relative proximity. This explanation is also supported by the considerable amount of sound samples corresponding to car repairs which were preceded by very few driving recordings.

6.1.6.2 *Comparison Between Classifiers*

In what follows, we will present and compare our classification results, using as measures the precision, recall and F_1 score. Figure 38 shows the performance of all combinations of feature extraction methods and classifiers that we used. Except the Naive Bayes algorithm, most classifiers provide relatively similar results. The

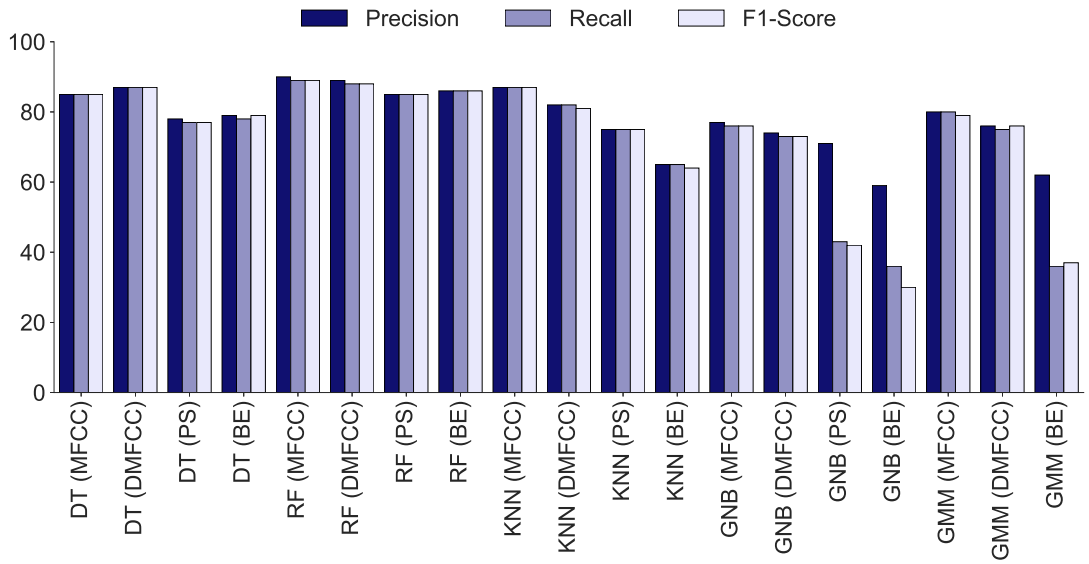


Figure 38: Comparison of the performances of all the combinations of feature types and classifiers

choice of feature extraction algorithm again plays an important role, MFCC and Delta MFCC being the best performing algorithms. The impact of the types of features used vary amongst classifiers. The overall best results by a small margin are obtain by MFCC and Random Forest.

6.1.6.3 *The Activity Classification in Detail*

Next, we will analyze in depth the best classification results, which were provided by Random Forest Classifier together with MFCC features. Table 7 shows the precision, recall, F_1 score and number of samples for each of the situations and for the overall classification. The overall precision of 90% could be achieved thanks to the distinct acoustic patterns of the situations.

Two apparently similar situations, repairing a car in the garage and repairing it outside of the garage are distinguished remarkably well too. While a technician might perform similar tasks when repairing a car, regardless of the location, when a car is repaired in the garage there will often be more than one technician working there. Thus, most of the recordings taking place inside the garage contain the overlapping sounds of more cars being repaired, while the recordings of cars being repaired on the street correspond to sounds of only one car being repaired.

The lowest precision among all activities was obtained for other activities performed in the garage (80%), besides repairs. Furthermore, this was not an activity of its own, but a cluster of activities which occurred much more rarely and thus were more difficult to learn for the classifier, given the differences within the same class and the scarceness of training data.

Figure 39 shows the confusion matrix for the activity classification. It can be noticed that for most activities the false positive and false negative rates are very low. The various activities carried out in the garage, besides car repairs cover an insignificant amount of time and are often confused with car repairs as both activities take

Class	Precision	Recall	F_1 score	Number of samples
Break in the garage	0.95	0.85	0.90	212
Meeting	0.87	0.95	0.91	724
Repair in the garage	0.86	0.90	0.88	298
Repair outside of the garage	0.95	0.84	0.89	390
Other work in the garage	0.8	0.21	0.33	19
Overall	0.90	0.89	0.89	1643

Table 7: Evaluation of the classification results of Random Forest and MFCC

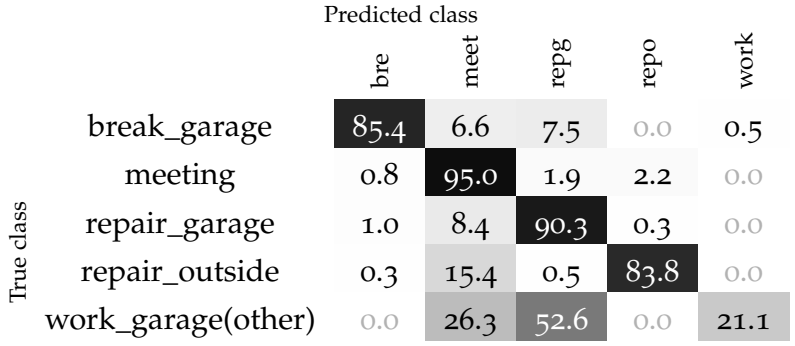


Figure 39: Classification results of Random Forest and MFCC in %, 1643 windows

place in the same space and there are often overlapping sounds of colleagues repairing other cars.

6.1.6.4 *Conclusions*

As our survey has shown, there is a need for a Q&A platform adapted to the needs of car technicians. Moreover, most car technicians are willing to help their colleagues, provided that the question does not interrupt them from certain actvities. Therefore, we have proposed and evaluated a user activity approach aimed specifically at car technicians. Our application relied on collecting and classifying audio recordings and achieved precision and recall values of about 90%. Therefore, out approach proved feasible for the given scenario.

6.2 A PLATFORM FOR PHONE MODE ADAPTATION

In the previous chapters and sections we have concerned ourselves with phone position and user activity detection and how to achieve excellent classification performance. Additionally, we have discussed possibilities for data acquisition and potential for user disturbance. In what follows we describe the *AutoMode* application which brings together most of the afore-presented methods and based on them adapts the phone mode. In a nutshell, AutoMode is a mobile application that listens for incoming phonecalls, notifications, and messages. When such an event is captured, certain sensors are sampled for a short time, then the samples are classified

and the phone mode of the phone is adapted according to an existing mapping table of phone positions and user activities to ideal phone modes.

We first present our design together with the corresponding decisions and then move on to the actual proof of feasibility. Our main constraint in this case is time, since we gather our data opportunistically, when a phonecall, message, or notification comes in. Thus, we need to classify the user's activity and phone position before the user can actually be disturbed by the incoming sound. This however does not allow us to compromise much on the machine learning mechanisms used, since misclassification will result in either missed notifications or in the disturbance of the user and potentially the surrounding people.

In what follows we explain our choice of sensors, then we present and explain our design decisions. We subsequently describe our data collection and data classification components. Finally, we present our experimental results that show the feasibility of using an opportunistic approach for determining phone position and user activity.

6.2.1 *Sensors*

Our main constraints, to reduce the adaptation time and to achieve a high classification accuracy, determine our criteria for selecting the sensors to use. Thus, we focus only on sensors that can yield very good results either for phone position classification alone, or at least can be used to determine if the phone lies within or outside of an enclosure. In the former category belong the microphone and accelerometer and in the latter the proximity and light sensor, as shown in Chapter 5. As determined experimentally, the ideal window size for our scenario is 4096 samples for audio and 8 samples for accelerometer. On Nexus 5 phones, the accelerometer requires 110 ms recording to ensure that at least 8 samples are collected. Since this is more than the microphone requires, we decided to set the window size to 110 ms. As we show in the following sections, this leads to a duration of 500 ms for the whole phone mode adaptation process. Therefore, this window duration increase (between 92 ms and 110 ms for audio) is a compromise we make given the excellent classification results it has proven to provide, but we will no further increase the window size duration for other sensors with limited utility. Table 8 shows the average number of samples obtained in 110 ms by microphone, accelerometer, light and proximity sensors, as well as the required time duration to get sufficient samples. The measurements were made on Nexus 5 phones. Given these times, we decided to rely on audio recordings and accelerometer samples for our proof of concept.

Sensor	#Samples in 110 ms Avg. over 50 tests	Min. rec. (ms) for sufficient samples
Audio	4851	92
Accelerometer	8.4	110
Light	2.6	400 (dep. on lighting changes)
Proximity	0	(dep. on lighting changes)

Table 8: Number of samples obtained for the considered sensors in 110 ms windows

6.2.2 *User Context*

In order to adapt phone mode, we classify two aspects: the phone position and user activity. The phone position class set consists of pocket, hand, bag, and table, as it did throughout Chapter 5. The considered user activities were selected in order to cover more of the user's day and not just the work day as considered in our user study. Thus, our activity set includes being in a meeting or discussion, taking a break at work, leisure time and commuting. Being in a meeting or discussion was deemed as an important aspect to setting the phone on "only lights" or silent mode by our user study respondents. Similarly ranked was being in a presentation, but given the similarity of these activities, both in regards to the sensor readings fingerprint and to the appropriate phone mode, we have decided to group these activities together. Working on a computer at work was deemed as moderately relevant for one phone mode and irrelevant for other phone modes so we decided to leave it out. Similarly we have left out physical activities around the office and making phonecalls which were rated as mostly irrelevant for all phone modes. We have added leisure activities and commuting to the set of regular activities.

6.2.3 *Phone Modes*

We have considered in our implementations the phone modes available on Samsung Galaxy Nexus and Nexus 5 phones, namely: Volume and Vibrate, Volume Only, Vibrate Only, and Silent. The "Volume Only" setting on notification works only if the app developer has turned off vibration for its notifications.[1]

Previously in this work we have used the terms: loud mode, normal mode, silent, and only lights. Thus, what we previously defined as silent mode corresponds to Vibrate Only and "only lights" maps to Nexus 5's Silent mode. The two remaining modes do not perfectly overlap, since the distinction between loud and normal mode is given by the volume of the ringtone. One can obviously adapt this setting as well, but we did not take this aspect into account in our implementation.

6.2.4 *Application Concept*

In the related work there are two main types of architecture for smartphone-based context-aware applications. One approach uses the phone just for collecting sensor data, which is subsequently sent to a server, where it is processed and classified. The classification result or another result based on the detected class is returned to the phone which takes the corresponding decision. This last step is optional for certain scenarios, e.g. when the server adapts its behavior towards the phone application based on contextual data.

The alternative approach carries out the whole process on the phone. Thus, data is collected and then classified on the phone, based on which the phone takes corresponding decisions.

Each of the two types of architectures presents several advantages and disadvantages. The first approach, which processes the data on the server, can use much more complex models given the increased computation power. Additionally, chang-

1 https://developer.android.com/reference/android/app/Notification.html#DEFAULT_VIBRATE

ing data models based on new input data or user feedback is quite easy, not involving the smartphone user at all. On the other hand, this approach relies on network quality and based on the user's location it can suffer from significant delays. On top of this, transferring the data away from the phone to a foreign entity presents increased privacy concerns and can hinder user acceptance.

Conversely, the approach that does all processing on the phone has lower privacy concerns and does not depend on network quality. However, the classification process requires more time and, depending on the scenario, one has to compromise on the classifier used and the size of the model. Additionally, changing the model requires user involvement or at the very least network connection and the download of a potentially large file together with the prior agreement of the user to this silent type of download.

For our scenario, a timely classification is crucial. We gather data opportunistically and only after classifying it adapt the phone mode, which creates a heightened potential for disturbance in case of low network quality and delays. For this reason, we opted for the phone-based data classification for our proof of concept.

6.2.5 *Application Description*

In what follows we present our application's main components: data collection and classification and provide a bird's eye view of their implementation.

6.2.5.1 *Data Collection*

Our app implements the notification listener service and telephony manager provided by the Android API to receive a callback for incoming notifications or calls. Once the callback is received, we start gathering audio and accelerometer samples for 110 ms. This recording duration corresponds to a single window and thus a single feature vector for each sensor. When the recording stops, we extract features from all recorded sensor data, generate an ARFF file, which is the Weka data format, and classify the activity, as shown in the following section.

6.2.5.2 *Data Classification*

All the methods used for our implementation were already used and discussed in our approaches from Chapter 5, so in what follows we shall only briefly mention them.

The preprocessing step involves data cleaning to remove noise. For audio we only carry out silence removal. The recording is not split into windows since recording length is the same as our chosen window length. Our previous test have shown the fitness of rectangular window functions, which refers to the windowing effect when no window function is applied. For the accelerometer readings, we use a standard data cleaning approach of filtering the values using a low-pass filter to remove the gravitational component.[2].

Next, we apply feature extraction to obtain an informative data representation. As explained in Chapter 5, for audio recordings we use the first 13 Mel Frequency

2 https://developer.android.com/guide/topics/sensors/sensors_motion.html#sensors-motion-accel

Cepstral Coefficients (MFCC) calculated over the entire audio recording as features. For accelerometer readings, we chose a window consisting of 8 samples for feature creation as it gave optimal results as shown earlier. For each value of a data point, we create the following features: mean, minimum, maximum, median, standard deviation, root mean square. These features are commonly used when dealing with numerical data. Additionally, we calculate the correlation for each pair of values as features. As an experiment, we used the classified phone position as feature for activity classification, assuming users would often have a specific preferred phone position for each activity.

After obtaining a feature representation of the raw data, we build a supervised machine learning model. We chose Decision Trees (DT), k-Nearest Neighbors (kNN) and Random Forest (RF) algorithms for modeling, since they performed well for the given classification tasks, as shown in the previous and current chapter. We used J48, the open source implementation of the state-of-the-art C4.5 decision trees algorithm. Despite having a high memory usage, we use kNN as it yielded good results for the current classification scenarios. We chose to use RF as it builds multiple Decision Trees for improving classifier performance.

We use the Weka machine learning library [40] to implement the J48 decision trees, kNN and RF algorithms as it is Java based and easily integrates in Android.

There are different options to train a classifier that has a good performance for multiple users. One option is to start with a small model and extend it based on user feedback. Thus, one would collect the data on the phone, classify it and adapt the phone mode, which in the beginning would be very much error-prone. However, one would also ask the user for the feedback and add the new data features together with the user-inputted class to the training file for improved classification at a later stage. This approach has the advantage of being personalized to each user and thus offering very good results for the specific user. However, it will only work for a single individual and will not be robust if the user activities change significantly. Additionally, it imposes the labeling overhead on the user and the erroneous initial results can lead to the user abandoning the application.

Another option is to collect sufficient data from multiple users to build a robust classification model and embed this in our app. This approach has the drawback of not being able to incorporate new user data as it is generated. An improvement is to incorporate the classifier model into the app, and collect user data on a server for updating the model. One can then release a new version of the application with an improved classifier.

We currently give the users the option to use either of the two approaches, which in practice results in either using only the existing model, or starting with the existing model and gradually adding new personal training data. The training data is stored in ARFF files. The initial file is generated based on data available from multiple users at the moment the app is released. In case the user opted to provide feedback to the classification results, the new feature vectors together with the corresponding class are added to the ARFF file.

6.2.5.3 *Phone Mode Adaptation*

As previously mentioned, in order to minimize phone mode adaptation time, we collect the shortest recording that still allows for a high classification performance.

This implies having a single window and thus a single feature vector for each sensor. Therefore, each sensor has just one vote for the user activity and phone position. In case of disagreement the current version of the app compares the two corresponding phone modes and errs on the safe side by picking the least disturbing one. The app has a predefined table that maps combinations of phone positions and user activities to phone modes. Since our user study showed no agreement amongst users regarding the appropriate phone modes, we offer the users the possibility to adjust these mappings based on personal preference.

6.2.6 *Feasibility of Opportunistic Data Acquisition*

As we have explained in Chapter 5, opportunistically gathering data and piggybacking audio and vibration sources to obtain pilot sequences has numerous advantages. Compared to duty cycling, we are guaranteed the currentness of the user context. Compared to continuous sampling we have significantly reduced battery impact and privacy concerns. Based on experimental analysis, we found that the microphone consumes an average of 2% battery every hour, and accelerometer consumes nearly 25% battery every hour when run continuously on Galaxy Nexus phones. Obtaining the pilot sequences for free allows us to avoid playing a specific pilot sequence and thus surely disturbing the user.

An important issue concerns the overall duration of the phone mode adaptation, from the moment the ringtone or notification event is triggered to the moment the phone mode has been appropriately adapted. This timing determines whether the incoming notification sound would be perceived by the user and whether this would cause a disturbance.

In what follows, we look in detail at the amount of time between a phonecall being received (the capture of a phonecall event) and the phone mode being adapted. Figures 40 and 41 show the app workflow together with the times required for each step for the two sensing modalities: 1) audio recordings and 2) accelerometer readings. We shall discuss the case of collecting and classifying audio recordings alone in more detail. The app workflow described below involves starting with a model built from the available data and incorporating all new data the user gives us feedback about then and hence takes greater processing time. For a 110 ms audio recording duration, processing takes an average of 536.9 ms. Hence, it takes 646.9 ms in total from the moment a notification is received until the phone mode is adapted. Once a notification is received, it takes 3.9 ms to enter the start recording function and 40.8 ms before starting to record audio. The time between starting to record audio and the end of recording audio is logged as 0 ms due to the asynchronous call to record audio. The record time of 110 ms is logged in the 117.7 ms between the end of recording audio and entering the stop recording runnable in the code. The 7.7 ms is taken by the Android API to release the audio recording resources. The 208.8 ms between end of moving the data to starting audio classification is due to creating MFCC features for the recorded audio. It takes 10.1 ms to classify audio data and 50.6 ms to change the phone mode. The workflows do not include the time taken by the user to manually set the correct label values.

A sound taking more than half a second can be heard, however given its duration its perceived loudness would be reduced compared to a normal audio clip. However,

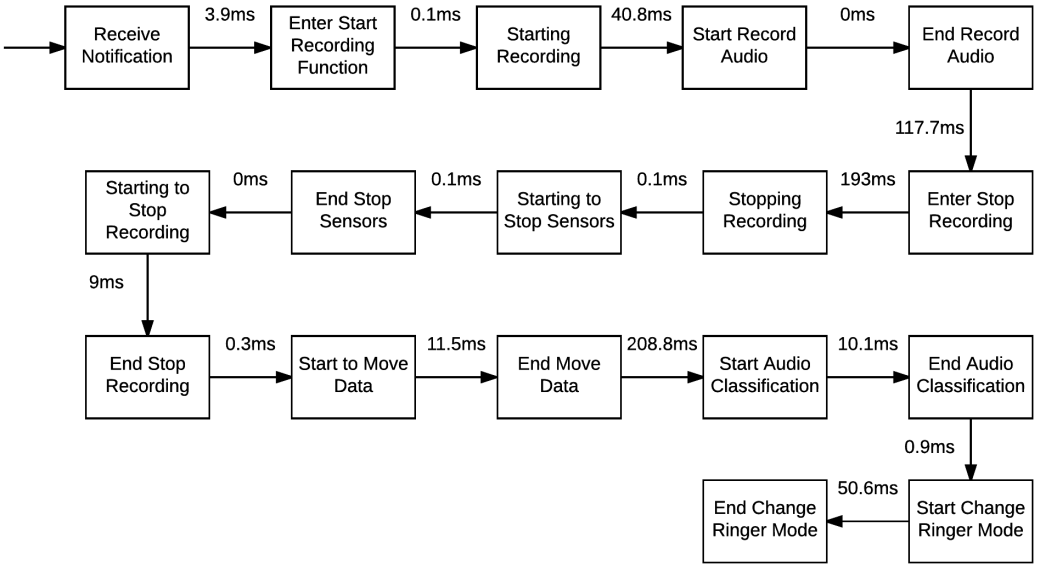


Figure 40: Processing time for the audio-based workflow

as shown by Figure 41, an approach based on accelerometer samples classification would take only about two thirds of the time required by audio classification, namely 358.9 ms. This would most likely be imperceivable by the user, however using it for all cases regardless of phone mode would involve being able to use active probing only for those phone modes which have vibration motor usage activated.

One option to further reduce the time needed for the complete process would be to optimize the feature extraction, since this takes a significant amount of time compared to other steps. The proof of concept uses an unoptimized, straight-forward implementation of the MFCC algorithm, which could either be profiled and optimized or reimplemented in a low-level language such as C which generally provides better performance.

6.2.7 *Performance*

In what follows, we will have a quick look at the preliminary results regarding the performance of the different classifiers and sensors. Given the extensive discussion of phone position classification provided in the previous chapter, we shall refer in what follows to user activity alone. Tables 9 and 10 present the activity-only classification results with one distinction: the latter considers phone position as a feature, whereas the former does not. One can notice the slight improvement brought by using phone position as feature, which seems to confirm our assumption that users have preferred phone positions for each activity.

We obtained the following performance listed in the table below using different classifiers using all the collected data to build a model on the PC. This is similar to using the final training file on the phone. All classifiers are implemented in Weka.

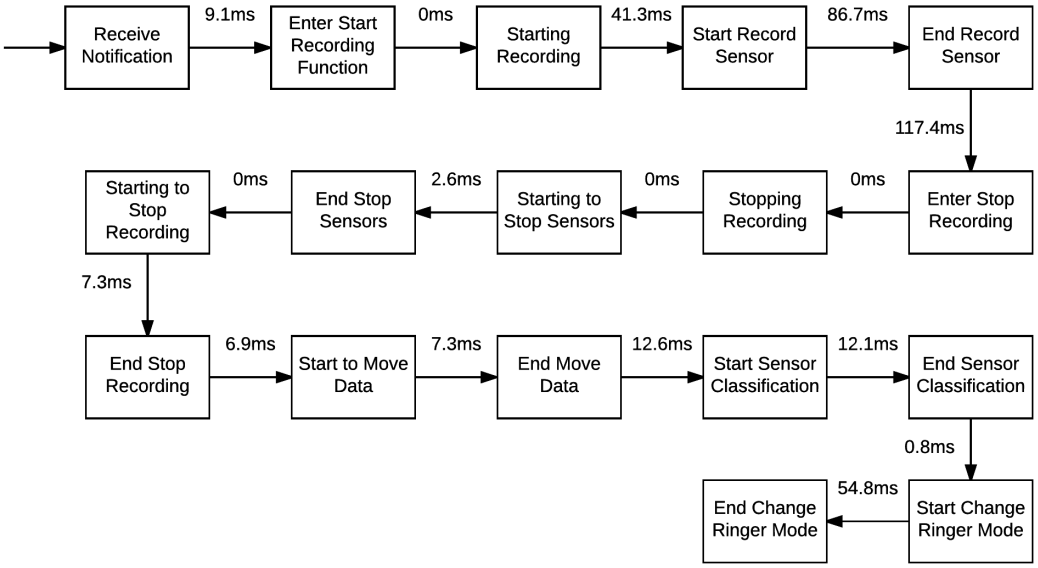


Figure 41: Processing time for the accelerometer-based workflow

Sensor	J48	kNN	RandomForest
Audio	75.54%	80.98%	83.15%
Accelerometer	86.90%	85.71%	86.90%

Table 9: Accuracy of audio recordings-based user activity classification without considering phone position as feature

6.2.8 *PoC Conclusions*

In the previous sections we have presented the main practical concerns with regards to the implementation of an automated phone mode adaptation app. Running the whole classification pipeline on the phone proved unproblematic, however the opportunistic adaptation of the phone mode would still require improvements before deployment in a real-life scenario. The opportunistic approach relying on collecting accelerometer readings while the vibration motor is triggered proved to be feasible for our scenario. In case current phone mode involves both sound and vibrations, an option would be to delay the playing of sounds until the phone mode has been adapted based on accelerometer samples classification results. However, in case that

Sensor	J48	kNN	RandomForest
Audio	81.52%	86.41%	84.24%
Accelerometer	86.90%	85.12%	88.09%

Table 10: Accuracy of audio recordings-based user activity classification while considering phone position as feature

the user uses a phone mode that relies solely on sound and not on the vibration motor our application would have to fall back on using the audio recordings classifications, which have a slightly more than half a second delay between the notification sound being played and the phone mode being adapted. This delay could lead to user disturbance and raises questions regarding the fitness of opportunistic audio classification for the given scenario. On the one hand, more research could be carried out regarding minimizing the data collection and feature extraction time, both from a theoretical and from a implementation perspective. For instance, from a theoretical perspective one could look into different feature extraction algorithms that require less computation time, while from a practical perspective one could compare the times required by the different libraries' implementations of the same feature extraction algorithms. On the other hand, the audio-based approach could be used in a scenario where the user still continues to adapt his phone mode and keeps the app as a safeguard against forgetfulness. In that latter case, a half a second sound might be a negligible disturbance.

7

CONCLUSIONS AND OUTLOOK

7.1 CONCLUSIONS

Despite the great developments in smartphone capabilities and applications, users still have to manually adapt their phone modes. Besides being a repetitive tasks that users have to remember to perform, failing to do so can result in either disturbances or missed calls and notifications.

In order to solve these issues, this thesis presented an approach for automatic phone mode adaptation based on phone position and user activity. We proposed three active probing approaches to classify phone position, which achieve accuracies in excess of 90%, two of which obtain their pilot sequences opportunistically. Since reasonable activity classification methods had already been developed, we simply incorporated them into our implementation. Finally, we presented an implementation relying on the afore-mentioned methods.

The challenges we targeted in this thesis were achieving a very high accuracy for phone position classification while creating minimal constraints or disturbances for the user and ensuring that the current classification result is not deprecated due to a very recent change in phone position.

Our first contribution is a user study to provide us with an understanding of the frequency with which users adapt their phone mode, together with the frequency of disturbances and missed phonecalls and notifications as a result from failing to do so. The findings underline the necessity of our solution, with more than 70% of the respondents adjusting their phone mode at least once or twice per day and more than 30% of the respondents carrying this out between three and five times per day. More importantly, more than 35% of the respondents cause a disturbance at least once per week and more than half of the respondents are at least once per week disturbed by other people not adjusting their phone mode. Additionally, we have shown that about 60% of the users are open to using an automatic phone mode adaptation application.

Our second contribution concerns the improvement of phone position classification accuracy. Thus, instead of classifying audio recordings of environment sounds alone, we proposed a method that plays certain sounds, e.g., Gaussian noise, and records them at the same time, then classifies the phone position. Our approach relies on very short sounds being played (~100 ms), in order to reduce the risk of user disturbance and minimize user privacy concerns. By doing so, we achieved accuracies in excess of 95%, whereas related work approaches reported accuracies of up to 84%.

Our third contribution proposes the opportunistic use of incoming ringtones and notification sounds as pilot sequences instead of playing them like in the previous approach. Thus, we can avoid duty cycling and can classify the phone position only when it is needed. In other words, the phone mode has to be the appropriate one only in case of an incoming notification, otherwise it is irrelevant. By avoiding duty

cycling we not only save battery, but also are sure of the currentness of the classified phone position. This approach still yields accuracies of over 90% for the majority of the default ringtones and notification sounds on Samsung Galaxy Nexus and Galaxy S3 phones. The slight decrease in accuracy compared to the second contribution is still offset by the afore-mentioned advantages of piggybacking notification sounds.

Since the previous approach does not apply for phones set to silent mode, our fourth contribution concerns the classification of accelerometer readings collected while the vibration motor is triggered. This method achieves accuracies of up to 99% and is obviously not affected by notification sounds. Furthermore, our evaluation has shown that user or environment movement have little impact on classification accuracy. This method is particularly appropriate for privacy-conscious users who find audio recordings, no matter how short, too intrusive.

Lastly, our fifth contribution is a proof of concept implementation of an automatic phone mode adaptation system. Our application waits for incoming phonecalls, messages, and notifications, opportunistically records incoming sounds or collects accelerometer readings, then classifies the phone position and user activity and adapts the phone mode. We adapted existing user activity classification methods to specific groups of users, each with their corresponding activity set. This implementation also addresses practical concerns regarding the feasibility of on the spot adaptation of the phone mode.

7.2 OUTLOOK

All the methods presented in this thesis, together with the further work opportunities depend heavily on the hardware settings of the phone as well as development of new wearables. Since we have limited ourselves to sensors built in the smartphones, new sensors incorporated in future smartphone models will provide now-unknown opportunities for the classification of the phone position, user activity, and generally for all other contextual information. Furthermore, the advent of wearables such as smartwatches that also provide notifications for the smartphone might render loud ringtones and notification sounds obsolete.

Future work could focus on identifying more fine-grained phone positions, for instance distinguishing between types of bags, or between jeans, skirt, or shirt pockets. This would provide additional insight with regards to how formal the situation is which would provide additional input for the phone mode decision. Thus, the user might carry the phone in their pocket and be in a meeting, but while wearing formal pants would suggest a more serious meeting, wearing jogging pants might point to a meeting with his friends where an incoming phonecall would cause no disturbance.

Additionally, the user's interruptibility might vary during a given activity, which in turns influences the appropriate phone mode.One can divide activities into their finer-grained components and aim to classify these components. For instance, the user might be in a presentation so normally not willing to read messages, however in the transition period between two speakers he might be willing to read his alerts, if provided in a discrete manner, e.g., by using the silent mode. New insight into these fine-grained components could be provided by using data from smartwatches and fitness bands.

BIBLIOGRAPHY

[1] Adamczyk, Piotr D. ; Bailey, Brian P.: If Not Now, When?: The Effects of Interruption at Different Moments Within Task Execution. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2004

[2] Azizyan, Martin ; Constandache, Ionut ; Roy Choudhury, Romit: SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In: *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, 2009

[3] Bailey, Brian P. ; Konstan, Joseph A. ; Carlis, John V.: Measuring the Effects of Interruptions on Task Performance in the User Interface. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, 2000

[4] Bailey, Brian P. ; Konstan, Joseph A. ; Carlis, John V.: The Effects of Interruptions on Task Performance, Annoyance, and Anxiety in the User Interface. In: *Proceedings of the 1st International Conference on Human-Computer Interaction (INTERACT)*, 2001

[5] Bao, Ling ; Intille, Stephen S.: Activity Recognition from User-Annotated Acceleration Data. In: *Proceedings of the Second International Conference on Pervasive Computing (PerCom)*, 2004

[6] Barwise, Jon ; Perry, John: The Situation Underground. In: *Stanford Working Papers in Semantics* 1 (1980)

[7] Bettini, Claudio ; Brdiczka, Oliver ; Henricksen, Karen ; Indulska, Jadwiga ; Nicklas, Daniela ; Ranganathan, Anand ; Riboni, Daniele: A Survey of Context Modelling and Reasoning Techniques. In: *Pervasive and Mobile Computing* 6 (2010), Nr. 2

[8] Bitkom: *Anteil der Smartphone-Nutzer in Deutschland in den Jahren 2012 bis 2016*. `https://de.statista.com/statistik/daten/studie/585883/umfrage/anteil-der-smartphone-nutzer-in-deutschland/`. – Online; accessed 01-December-2017

[9] Böhmer, Matthias ; Hecht, Brent ; Schöning, Johannes ; Krüger, Antonio ; Bauer, Gernot: Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on Mobile Application Usage. In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, 2011

[10] Breiman, Leo: Random Forests. In: *Machine learning* 45 (2001), Nr. 1

[11] Center, Pew R.: *U.S. Smartphone Use in 2015*. `http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/`. – Online; accessed 01-December-2017

[12] Chen, Jianfeng ; Kam, Alvin H. ; Zhang, Jianmin ; Liu, Ning ; Shue, Louis: Bathroom Activity Monitoring Based on Sound. In: *Pervasive Computing*. 2005

[13] Chen, Ke-Yu ; Ashbrook, Daniel ; Goel, Mayank ; Lee, Sung-Hyuck ; Patel, Shwetak: AirLink: Sharing Files Between Multiple Devices Using In-air Gestures. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[14] Cho, Jungchan ; Hwang, Inhwan ; Oh, Songhwai: Vibration-Based Surface Recognition for Smartphones. In: *Proceedings of the 2012 IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2012

[15] Choe, Eun K. ; Lee, Bongshin ; Kay, Matthew ; Pratt, Wanda ; Kientz, Julie A.: SleepTight: Low-burden, Self-monitoring Technology for Capturing and Reflecting on Sleep Behaviors. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[16] Choi, Woohyeok ; Oh, Jeungmin ; Park, Taiwoo ; Kang, Seongjun ; Moon, Miri ; Lee, Uichin ; Hwang, Inseok ; Song, Junehwa: MobyDick: An Interactive Multi-swimmer Exergame. In: *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2014

[17] Chon, Yohan ; Lane, Nicholas D. ; Li, Fan ; Cha, Hojung ; Zhao, Feng: Automatically Characterizing Places with Opportunistic Crowdsensing Using Smartphones. In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 2012

[18] Choudhury, Tanzeem ; Consolvo, Sunny ; Harrison, Beverly ; Hightower, Jeffrey ; LaMarca, Anthony ; LeGrand, Louis ; Rahimi, Ali ; Rea, Adam ; Borriello, Gaetano ; Hemingway, Bruce ; Klasnja, Predrag ; Koscher, Karl ; Landay, James A. ; Lester, Jonathan ; Wyatt, Danny ; Haehnel, Dirk: The Mobile Sensing Platform: An Embedded System for Activity Recognition. In: *IEEE Pervasive Computing* 7 (2008), Nr. 2

[19] Consolvo, Sunny ; McDonald, David W. ; Toscos, Tammy ; Chen, Mike Y. ; Froehlich, Jon ; Harrison, Beverly ; Klasnja, Predrag ; LaMarca, Anthony ; LeGrand, Louis ; Libby, Ryan u. a.: Activity Sensing in the Wild: A Field Trial of UbiFit Garden. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008

[20] Constandache, Ionut ; Bao, Xuan ; Azizyan, Martin ; Choudhury, Romit R.: Did You See Bob?: Human Localization Using Mobile Phones. In: *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2010

[21] Cui, Yanqing ; Jan, Chipchase ; Fumiko, Ichikawa: A Cross Culture Study on Phone Carrying and Physical Personalization. In: *Proceedings of the 2nd International Conference on Usability and Internationalization*, 2007

[22] Czerwinski, Mary ; Cutrell, Edward ; Horvitz, Eric: Instant Messaging and Interruption: Influence of Task Type on Performance. In: *Proceedings of the 12th Australian Conference on Human-Computer Interaction (OZCHI)*, 2000

[23] Després, Lena ; Rensing, Christoph ; Diaconita, Irina: Konzeption, Durchführung und Ergebnisse einer Usability-Studie zu einem Q&A System für mobiles Lernen von Servicetechnikern. In: *Die 12. E-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. (DeLFI)*, 2014

[24] Dey, Anind K.: *Providing Architectural Support for Building Context-aware Applications*, Georgia Institute of Technology, Dissertation, 2000

[25] Diaconita, Irina ; Reinhardt, Andreas ; Christin, Delphine ; Rensing, Christoph: Bleep Bleep! Determining Smartphone Locations by Opportunistically Recording Notification Sounds. In: *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2014

[26] Diaconita, Irina ; Reinhardt, Andreas ; Christin, Delphine ; Rensing, Christoph: Inferring Smartphone Positions based on Collecting the Environment's Response to Vibration Motor Actuation. In: *Proceedings of the ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM) Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet)*, 2015

[27] Diaconita, Irina ; Reinhardt, Andreas ; Englert, Frank ; Christin, Delphine ; Steinmetz, Ralf: Do You Hear What I Hear? Using Acoustic Probing to Detect Smartphone Locations. In: *Proceedings of the 12th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2014

[28] Diaconita, Irina ; Rensing, Christoph ; Tittel, Stephan: Getting the Information You Need, When You Need It: A Context-aware Q&A System for Collaborative Learning. In: *Proceedings of the 9th European Conference on Technology Enhanced Learning (EC-TEL)*, 2014

[29] Diaconita, Irina ; Rensing, Christoph ; Tittel, Stephan u. a.: Context-aware Question and Answering for Community-based Learning. In: *Die 11. E-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. (DeLFI)*, 2013

[30] Dutz, Tim: *Pervasive Behavior Interventions – Using Mobile Devices for Overcoming Barriers for Physical Activity*. Darmstadt, Technische Universität Darmstadt, Dissertation, 11 2016

[31] Easton, Valerie J. ; McColl, John H.: *Statistics Glossary*. `http://www.stats.gla.ac.uk/steps/glossary/index.html`. 1997. – Online; accessed 01-December-2017

[32] Englert, Frank ; Diaconita, Irina ; Reinhardt, Andreas ; Alhamoud, Alaa ; Meister, Richard ; Backert, Lucas ; Steinmetz, Ralf: Reduce the Number of Sensors: Sensing Acoustic Emissions to Estimate Appliance Energy Usage. In: *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, 2013

[33] Eronen, A. J. ; Peltonen, V. T. ; Tuomi, J. T. ; Klapuri, A. P. ; Fagerlund, S. ; Sorsa, T. ; Lorho, G. ; Huopaniemi, J.: Audio-based context recognition. In: *IEEE Transactions on Audio, Speech, and Language Processing* 14 (2006)

[34] Eronen, Antti: Comparison of Features for Musical Instrument Recognition. In: *Proceedings of the IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, 2001

[35] Fisher, Robert ; Simmons, Reid: Smartphone Interruptibility Using Density-weighted Uncertainty Sampling with Reinforcement Learning. In: *Proceedings of the 10th International Conference on Machine Learning and Applications and Workshops (ICMLA)*, 2011

[36] Fisher, Ronald A.: The use of multiple measurements in taxonomic problems. In: *Annals of Eugenics* 7 (1936), Nr. 2

[37] Fogarty, James ; Hudson, Scott E. ; Atkeson, Christopher G. ; Avrahami, Daniel ; Forlizzi, Jodi ; Kiesler, Sara ; Lee, Johnny C. ; Yang, Jie: Predicting Human Interruptibility with Sensors. In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 12 (2005), Nr. 1

[38] Fujinami, Kaori ; Kouchi, Satoshi: Recognizing a Mobile Phone's Storing Position as a Context of a Device and a User. In: *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. 2013

[39] General Services Administration Information Technology Service: *Telecommunications: Glossary of Telecommunication Terms*. 1996

[40] Hall, Mark ; Frank, Eibe ; Holmes, Geoffrey ; Pfahringer, Bernhard ; Reutemann, Peter ; Witten, Ian H.: The WEKA Data Mining Software: An Update. In: *SIGKDD Explor. Newsl.* 11 (2009), November, Nr. 1

[41] Hao, Tian ; Xing, Guoliang ; Zhou, Gang: iSleep: Unobtrusive Sleep Quality Monitoring Using Smartphones. In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2013

[42] Hao, Tian ; Xing, Guoliang ; Zhou, Gang: RunBuddy: A Smartphone System for Running Rhythm Monitoring. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[43] Hardy, Sandro ; El Saddik, Abdulmotaleb ; Göbel, Stefan ; Steinmetz, Ralf: Context Aware Serious Games Framework for Sport and Health. In: *Proceedings of the 2011 IEEE International Symposium on Medical Measurements and Applications*, 2011

[44] Harr, Rikard ; Kaptelinin, Victor: Interrupting or Not: Exploring the Effect of Social Context on Interrupters' Decision Making. In: *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design (NordiCHI)*, 2012

[45] Hemminki, Samuli ; Nurmi, Petteri ; Tarkoma, Sasu: Accelerometer-based Transportation Mode Detection on Smartphones. In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, 2013

[46] Henricksen, Karen ; Indulska, Jadwiga: Developing Context-aware Pervasive Computing Applications: Models and Approach. In: *Pervasive and Mobile Computing* 2 (2006), Nr. 1

[47] Ho, Joyce ; Intille, Stephen S.: Using Context-aware Computing to Reduce the Perceived Burden of Interruptions from Mobile Devices. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, 2005

[48] Hong, Dezhi ; Nirjon, Shahriar ; Stankovic, John A. ; Stone, David J. ; Shen, Guobin: Poster Abstract: a Mobile-Cloud Service for Physiological Anomaly Detection on Smartphones. In: *Proceedings of the 12th International Conference on Information Processing in Sensor Networks*, 2013

[49] Howell, David C.: *Fundamental Statistics for the Behavioral Sciences (7th ed.)*. Belmont, CA : Nelson Education, 2016

[50] Huang, Yan ; Powell, Jason W.: Detecting Regions of Disequilibrium in Taxi Services Under Uncertainty. In: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, 2012

[51] Hung, Hayley ; Englebienne, Gwenn ; Kools, Jeroen: Classifying Social Actions with a Single Accelerometer. In: *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2013

[52] Huýnh, Tâm ; Blanke, Ulf ; Schiele, Bernt: Scalable Recognition of Daily Activities with Wearable Sensors. In: *Proceedings of the International Symposium on Location and Context-Awareness*, 2007

[53] Iqbal, Shamsi T. ; Bailey, Brian P.: Leveraging Characteristics of Task Structure to Predict the Cost of Interruption. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2006

[54] Iqbal, Shamsi T. ; Bailey, Brian P.: Understanding and Developing Models for Detecting and Differentiating Breakpoints During Interactive Tasks. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2007

[55] Iqbal, Shamsi T. ; Bailey, Brian P.: Effects of Intelligent Notification Management on Users and Their Tasks. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008

[56] Jun, Junghyun ; Gu, Yu ; Cheng, Long ; Lu, Banghui ; Sun, Jun ; Zhu, Ting ; Niu, Jianwei: Social-Loc: Improving Indoor Localization with Social Sensing. In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2013

[57] Kannan, Pravein G. ; Venkatagiri, Seshadri P. ; Chan, Mun C. ; Ananda, Akhihebbal L. ; Peh, Li-Shiuan: Low Cost Crowd Counting Using Audio Tones. In: *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2012

[58] Keally, Matthew ; Zhou, Gang ; Xing, Guoliang ; Wu, Jianxin ; Pyles, Andrew: PBN: Towards Practical Activity Recognition Using Smartphone-based Body Sensor Networks. In: *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2011

[59] Kim, Yungeun ; Shin, Hyojeong ; Cha, Hojung: Smartphone-based Wi-Fi Pedestrian-tracking System Tolerating the RSS Variance Problem. In: *Proceedings of the 10th International Conference on Pervasive Computing and Communications (PerCom)*, 2012

[60] Korpela, Joseph ; Miyaji, Ryosuke ; Maekawa, Takuya ; Nozaki, Kazunori ; Tamagawa, Hiroo: Evaluating Tooth Brushing Performance with Smartphone Sound Data. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[61] Kranz, Matthias ; Möller, Andreas ; Hammerla, Nils ; Diewald, Stefan ; Plötz, Thomas ; Olivier, Patrick ; Roalter, Luis: The Mobile Fitness Coach: Towards Individualized Skill Assessment Using Personalized Mobile Devices. In: *Pervasive and Mobile Computing* 9 (2013), Nr. 2

[62] Kropff, Matthias: *Sensor-basiertes Monitoring zur kontextsensitiven Unterstützung von Wissensarbeit*. Darmstadt, Technische Universität Darmstadt, Dissertation, 12 2010

[63] Kwapisz, Jennifer R. ; Weiss, Gary M. ; Moore, Samuel A.: Activity Recognition Using Cell Phone Accelerometers. In: *ACM SIGKDD Explorations Newsletter* 12 (2011), Nr. 2

[64] Lane, Nicholas D. ; Lu, Hong ; Eisenman, Shane B. ; Campbell, Andrew T.: Cooperative Techniques Supporting Sensor-Based People-Centric Inferencing. In: *Proceedings of the 6th International Conference on Pervasive Computing*, 2008

[65] Lane, Nicholas D. ; Miluzzo, Emiliano ; Lu, Hong ; Peebles, Daniel ; Choudhury, Tanzeem ; Campbell, Andrew T.: A Survey of Mobile Phone Sensing. In: *IEEE Communications Magazine* 48 (2010), Nr. 9

[66] Lane, Nicholas D. ; Pengyu, Li ; Zhou, Lin ; Zhao, Feng: Connecting Personal-scale Sensing and Networked Community Behavior to Infer Human Activities. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[67] Larson, Eric C. ; Lee, TienJui ; Liu, Sean ; Rosenfeld, Margaret ; Patel, Shwetak N.: Accurate and Privacy Preserving Cough Sensing Using a Low-cost Microphone. In: *Proceedings of the 13th International Conference on Ubiquitous Computing*, 2011

[68] Lee, Haechan ; Moon, Miri ; Park, Taiwoo ; Hwang, Inseok ; Lee, Uichin ; Song, Junehwa: Dungeons & Swimmers: Designing an Interactive Exergame for Swimming. In: *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication (UbiComp)*, 2013

[69] Li, Qi ; Zheng, Jinsong ; Tsai, Augustine ; Zhou, Qiru: Robust Endpoint Detection and Energy Normalization for Real-time Speech and Speaker Recognition. In: *IEEE Transactions on Speech and Audio Processing* 10 (2002), Nr. 3

[70] Li, Qiang ; Stankovic, John A. ; Hanson, Mark A. ; Barth, Adam T. ; Lach, John ; Zhou, Gang: Accurate, Fast Fall Detection Using Gyroscopes and

Accelerometer-Derived Posture Information. In: *Proceedings of the 6th International Workshop on Wearable and Implantable Body Sensor Networks*, 2009

[71] Liao, Lin ; Fox, Dieter ; Kautz, Henry: Extracting Places and Activities from GPS traces Using Hierarchical Conditional Random Fields. In: *The International Journal of Robotics Research* 26 (2007), Nr. 1

[72] Liu, Jiayang ; Zhong, Lin ; Wickramasuriya, Jehan ; Vasudevan, Venu: uWave: Accelerometer-based Personalized Gesture Recognition and its Applications. In: *Pervasive and Mobile Computing* 5 (2009), Nr. 6

[73] Logan, Beth u. a.: Mel Frequency Cepstral Coefficients for Music Modeling. In: *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 2000

[74] Lu, Hong ; Frauendorfer, Denise ; Rabbi, Mashfiqui ; Mast, Marianne S. ; Chittaranjan, Gokul T. ; Campbell, Andrew T. ; Gatica-Perez, Daniel ; Choudhury, Tanzeem: StressSense: Detecting Stress in Uconstrained Acoustic Environments Using Smartphones. In: *Proceedings of the ACM Conference on Ubiquitous Computing*, 2012

[75] Lu, Hong ; Pan, Wei ; Lane, Nicholas D. ; Choudhury, Tanzeem ; Campbell, Andrew T.: SoundSense: Scalable Sound Sensing for People-centric Applications on Mobile Phones. In: *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2009

[76] Luo, Chengwen ; Chan, Mun C.: SocialWeaver: Collaborative Inference of Human Conversation Networks Using Smartphones. In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2013

[77] Mahdaviani, Maryam ; Choudhury, Tanzeem: Fast and Scalable Training of Semi-Supervised CRFs with Application to Activity Recognition. In: *Advances in Neural Information Processing Systems*. 2008

[78] Mallick, Satya: *Bias-Variance Tradeoff in Machine Learning*. `https://www.learnopencv.com/bias-variance-tradeoff-in-machine-learning/`. 2017. – Online; accessed 01-December-2017

[79] Manning, Christopher D. ; Raghavan, Prabhakar ; Schütze, Hinrich: *Introduction to Information Retrieval*. Cambride : Cambridge University Press, 2008

[80] Maurer, U. ; Smailagic, A. ; Siewiorek, D.P. ; Deisher, M.: Activity Recognition and Monitoring Using Multiple Sensors on Different Body Positions. In: *Proceedings of the International Workshop onWearable and Implantable Body Sensor Networks*, 2006

[81] Mazilu, Sinziana ; Blanke, Ulf ; Hardegger, Michael ; Troster, Gerhard ; Gazit, Eran ; Dorfman, Moran ; Hausdorff, Jeffrey M.: GaitAssist: A Wearable Assistant for Gait Training and Rehabilitation in Parkinson's Disease. In: *Proceedings of the 12th International Conference on Pervasive Computing and Communications (PerCom) Workshops*, 2014

[82] McDonald, John H.: *Handbook of Biological Statistics (3rd ed.)*. Baltimore, Maryland : Sparky House Publishing, 2014

[83] Mehrotra, Abhinav ; Musolesi, Mirco ; Hendley, Robert ; Pejovic, Veljko: Designing Content-driven Intelligent Notification Mechanisms for Mobile Applications. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[84] Mesaros, Annamaria ; Heittola, Toni ; Eronen, Antti ; Virtanen, Tuomas: Acoustic Event Detection in Real Life Recordings. In: *Proceedings of the 18th European Signal Processing Conference*, 2010

[85] Miluzzo, Emiliano ; Lane, Nicholas D. ; Eisenman, Shane B. ; Campbell, Andrew T.: CenceMe–Injecting Sensing Presence into Social Networking Applications. In: *Proceedings of the Second European Conference on Smart Sensing and Context*, 2007

[86] Miluzzo, Emiliano ; Lane, Nicholas D. ; Fodor, Kristóf ; Peterson, Ronald ; Lu, Hong ; Musolesi, Mirco ; Eisenman, Shane B. ; Zheng, Xiao ; Campbell, Andrew T.: Sensing Meets Mobile Social Networks: the Design, Implementation and Evaluation of the CenceMe Application. In: *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2008

[87] Miluzzo, Emiliano ; Papandrea, Michela ; Lane, Nicholas D. ; Lu, Hong ; Campbell, Andrew T.: Pocket, Bag, Hand, etc.-Automatically Detecting Phone Context Through Discovery. In: *Proceedings of the ACM International Workshop on Sensing Applications on Mobile Phones*, 2010

[88] Miluzzo, Emiliano and Cornelius, Cory T. and Ramaswamy, Ashwin and Choudhury, Tanzeem and Liu, Zhigang and Campbell, Andrew T.: Darwin Phones: The Evolution of Sensing and Inference on Mobile Phones. In: *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2010

[89] Minnen, David ; Starner, Thad ; Essa, Irfan ; Isbell, Charles: Discovering Characteristic Actions from On-body Sensor Data. In: *Proceedings of the 10th IEEE International Symposium on Wearable Computers*, 2006

[90] Mitchell, Edmond ; Monaghan, David ; O'Connor, Noel E.: Classification of Sporting Activities Using Smartphone Accelerometers. In: *Sensors* 13 (2013), Nr. 4

[91] Mitchell, Tom M.: *Machine Learning*. New York : McGraw-Hill, 1997

[92] Molau, S. ; Pitz, M. ; Schluter, R. ; Ney, H.: Computing Mel-Frequency Cepstral Coefficients on the Power Spectrum. In: *Proceedings the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001

[93] Mun, Min ; Reddy, Sasank ; Shilton, Katie ; Yau, Nathan ; Burke, Jeff ; Estrin, Deborah ; Hansen, Mark ; Howard, Eric ; West, Ruth ; Boda, Péter: PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research. In: *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2009

[94] Muscillo, Rossana ; Schmid, Maurizio ; Conforto, Silvia ; D'Alessio, Tommaso: An Adaptive Kalman-based Bayes Estimation Technique to Classify Locomotor Activities in Young and Elderly Adults Through Accelerometers. In: *Medical Engineering & Physics* 32 (2010), Nr. 8

[95] Nandugudi, Anandatirtha ; Ki, Taeyeon ; Nuessle, Carl ; Challen, Geoffrey: PocketParker: Pocketsourcing Parking Lot Availability. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[96] National Instruments Corporation: *Understanding FFTs and Windowing*. `http://download.ni.com/evaluation/pxi/Understanding%20FFTs%20and%20Windowing.pdf`. – Online; accessed 01-December-2017

[97] Nirjon, Shahriar ; Dickerson, Robert F. ; Asare, Philip ; Li, Qiang ; Hong, Dezhi ; Stankovic, John A. ; Hu, Pan ; Shen, Guobin ; Jiang, Xiaofan: Auditeur: A Mobile-Cloud Service Platform for Acoustic Event Detection on Smartphones. In: *Proceedings of the The 11th International Conference on Mobile Systems, Applications, and Services*, 2013

[98] Nirjon, Shahriar ; Dickerson, Robert F. ; Li, Qiang ; Asare, Philip ; Stankovic, John A. ; Hong, Dezhi ; Zhang, Ben ; Jiang, Xiaofan ; Shen, Guobin ; Zhao, Feng: MusicalHeart: a *Hearty* Way of Listening to Music. In: *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, 2012

[99] Nishida, Kyosuke ; Toda, Hiroyuki ; Kurashima, Takeshi ; Suhara, Yoshihiko: Probabilistic Identification of Visited Point-of-interest for Personalized Automatic Check-in. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[100] Oliver, Nuria ; Horvitz, Eric: A Comparison of HMMs and Dynamic Bayesian Networks for Recognizing Office Activities. In: *Proceedings of the 10th International Conference on User Modeling (UM)*, 2005

[101] Park, Taiwoo ; Hwang, Inseok ; Lee, Uichin ; Lee, Sunghoon I. ; Yoo, Chungkuk ; Lee, Youngki ; Jang, Hyukjae ; Choe, Sungwon P. ; Park, Souneil ; Song, Junehwa: ExerLink: Enabling Pervasive Social Exergames with Heterogeneous Exercise Devices. In: *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2012

[102] Parkka, Juha ; Ermes, Miikka ; Korpipaa, Panu ; Mantyjarvi, Jani ; Peltola, Johannes ; Korhonen, Ilkka: Activity Classification Using Realistic Data From Wearable Sensors. In: *IEEE Transactions on Information Technology in Biomedicine* 10 (2006), Nr. 1

[103] Patterson, Donald J. ; Fox, Dieter ; Kautz, Henry ; Philipose, Matthai: Fine-grained Activity Recognition by Aggregating Abstract Object Usage. In: *Proceedings of the 9th IEEE International Symposium on Wearable Computers (ISWC'05)*, 2005

[104] Pedregosa, Fabian ; Varoquaux, Gael ; Gramfort, Alexandre ; Michel, Vincent ; Thirion, Bertrand ; Grisel, Olivier ; Blondel, Mathieu ; Prettenhofer,

Peter ; Weiss, Ron ; Dubourg, Vincent ; Vanderplas, Jake ; Passos, Alexandre ; Cournapeau, David ; Brucher, Matthieu ; Perrot, Matthieu ; Duchesnay, Edouard: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011)

[105] Peebles, Daniel ; Lu, Hong ; Lane, Nicholas D. ; Choudhury, Tanzeem ; Campbell, Andrew T.: Community-Guided Learning: Exploiting Mobile Sensor Users to Model Human Behavior. In: *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, 2010

[106] Pejovic, Veljko ; Musolesi, Mirco: InterruptMe: Designing Intelligent Prompting Mechanisms for Pervasive Applications. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[107] Peltonen, Vesa ; Tuomi, Juha ; Klapuri, A. ; Huopaniemi, Jyri ; Sorsa, Timo: Computational Auditory Scene Recognition. In: *Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2002

[108] Philipose, Matthai ; Fishkin, Kenneth P. ; Perkowitz, Mike ; Patterson, Donald J. ; Fox, Dieter ; Kautz, Henry ; Hahnel, Dirk: Inferring Activities From Interactions With Objects. In: *IEEE Pervasive Computing* 3 (2004), Nr. 4

[109] Philipp, Damian ; Baier, Patrick ; Dibak, Christoph ; Dürr, Frank ; Rothermel, Kurt ; Becker, Susanne ; Peter, Michael ; Fritsch, Dieter: Mapgenie: Grammar-enhanced Indoor Map Construction from Crowd-sourced Data. In: *Proceedings of the 12th IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2014

[110] Pielot, Martin: Large-scale Evaluation of Call-availability Prediction. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[111] Pielot, Martin ; Dingler, Tilman ; Pedro, Jose S. ; Oliver, Nuria: When Attention is Not Scarce - Detecting Boredom from Mobile Phone Usage. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[112] Pielot, Martin ; Oliveira, Rodrigo de ; Kwak, Haewoon ; Oliver, Nuria: Didn't You See My Message?: Predicting Attentiveness to Mobile Instant Messages. In: *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems (CHI)*, 2014

[113] Poppinga, Benjamin ; Heuten, Wilko ; Boll, Susanne: Sensor-based identification of opportune moments for triggering notifications. In: *IEEE Pervasive Computing* 13 (2014), Nr. 1

[114] Provost, Foster J. ; Fawcett, Tom ; Kohavi, Ron: The Case Against Accuracy Estimation for Comparing Induction Algorithms. In: *Proceedings of the 25th International Conference on Machine Learning (ICML)*, 1998

[115] Radu, Valentin ; Katsikouli, Panagiota ; Sarkar, Rik ; Marina, Mahesh K.: A Semi-supervised Learning Approach for Robust Indoor-outdoor Detection with Smartphones. In: *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2014

[116] Ramírez, Javier ; Segura, José C. ; Benítez, Carmen ; De La Torre, Ángel ; Rubio, Antonio: Efficient Voice Activity Detection Algorithms Using Long-term Speech Information. In: *Speech Communication* 42 (2004), Nr. 3

[117] Rana, Rajib K. ; Chou, Chun T. ; Kanhere, Salil S. ; Bulusu, Nirupama ; Hu, Wen: Ear-phone: an End-to-end Participatory Urban Noise Mapping System. In: *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010

[118] Rees, D. G.: *Essential Statistics (4th ed.)*. Boca Raton : Chapman and Hall/CRC, 2000

[119] Rensing, Christoph ; Diaconita, Irina: A Q&A System Considering Employees' Willingness to Help Colleagues and to Look for Help in Different Workplace-Related Situations: An Analysis in the Automotive Sector. In: *Proceedings of the 14th International IEEE Conference on Advanced Learning Technologies (ICALT)*, 2014

[120] Reynolds, Douglas: *Gaussian Mixture Models*. In: *Encyclopedia of Biometrics*. Boston, MA : Springer US, 2015

[121] Roberts, Stephen: *Lecture Notes: Signal Processing and Filter Design*. `http://www.robots.ox.ac.uk/~sjrob/Teaching/sp_course.html`. 2003. – Online; accessed 01-December-2017

[122] Rokach, Lior ; Maimon, Oded: *Data Mining with Decision Trees: Theory and Applications (2nd ed.)*. Hackensack, New Jersey : World Scientific, 2014

[123] Rossi, Mirco ; Feese, Sebastian ; Amft, Oliver ; Braune, Nils ; Martis, Sandro ; Troster, Gerhardt: AmbientSense: A Real-Time Ambient Sound Recognition System for Smartphones. In: *Proceedings of the 11th International Conference on Pervasive Computing and Communications (PerCom)*, 2013

[124] Russel, Stuart ; Norvig, Peter: *Artificial Intelligence: A Modern Approach (3rd ed.)*. Upper Saddle River, NJ : Prentice Hall Press, 1997

[125] Sankaran, Kartik ; Zhu, Minhui ; Guo, Xiang F. ; Ananda, Akkihebbal L. ; Chan, Mun C. ; Peh, Li-Shiuan: Using Mobile Phone Barometer for Low-power Transportation Context Detection. In: *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2014

[126] Sarker, Hillol ; Sharmin, Moushumi ; Ali, Amin A. ; Rahman, Md. M. ; Bari, Rummana ; Hossain, Syed M. ; Kumar, Santosh: Assessing the Availability of Users to Engage in Just-in-time Intervention in the Natural Environment. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[127] Scellato, Salvatore ; Musolesi, Mirco ; Mascolo, Cecilia ; Latora, Vito ; Campbell, Andrew T.: NextPlace: A Spatio-temporal Prediction Framework for Pervasive Systems. In: *Pervasive Computing*. Springer Berlin Heidelberg, 2011

[128] Schilit, Bill N. ; Theimer, Marvin M.: Disseminating Active Map Information to Mobile Hosts. In: *IEEE Network* 8 (1994), Nr. 5

[129] Schmidt, Albrecht ; Aidoo, Kofi A. ; Takaluoma, Antti ; Tuomela, Urpo ; Van Laerhoven, Kristof ; Velde, Walter Van de: Advanced Interaction in Context. In: *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, 1999

[130] Schmidt, Albrecht ; Beigl, Michael ; Gellersen, Hans-W: There is More to Context than Location. In: *Computers & Graphics* 23 (1999), Nr. 6

[131] Schmitt, Johannes: *Anpassungsfähige Kontextbestimmung zur Unterstützung von Kommunikationsdiensten*. Darmstadt, Technische Universität Darmstadt, Dissertation, 12 2009

[132] Schneider, Jeff ; Moore, Andrew: *A Locally Weighted Learning Tutorial using Vizier 1.0*. `https://www.cs.cmu.edu/~schneide/tut5/node42.html`. 1997. – Online; accessed 01-December-2017

[133] Seltman, Howard J.: *Experimental Design and Analysis*. `http://www.stat.cmu.edu/~hseltman/309/Book/Book.pdf`. 2012. – Online; accessed 01-December-2017

[134] Sen, Rijurekha ; Lee, Youngki ; Jayarajah, Kasthuri ; Misra, Archan ; Balan, Rajesh K.: GruMon: Fast and Accurate Group Monitoring for Heterogeneous Urban Spaces. In: *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2014

[135] Shafer, Ilari: *Learning Location from Vibration*. `http://www.mrcaps.com/proj/LocationVibration/files/vibration-ishafer-paper.pdf`. 2013. – Online; accessed 01-December-2017

[136] Shaw Jr, William M. ; Burgin, Robert ; Howell, Patrick: Performance Standards and Evaluations in IR Test Collections: Cluster-based Retrieval Models. In: *Information Processing & Management* 33 (1997), Nr. 1

[137] Shirazi, S. N. ; Gouglidis, A. ; Syeda, K. N. ; Simpson, S. ; Mauthe, A. ; Stephanakis, I. M. ; Hutchison, D.: Evaluation of Anomaly Detection techniques for SCADA communication resilience. In: *Proceedings of the 2016 Resilience Week (RWS)*, 8 2016

[138] Siewiorek, Daniel ; Krause, Andreas ; Moraveji, Neema ; Smailagic, Asim ; Furukawa, Junichi ; Reiger, Kathryn ; Wong, Fei L. ; Shaffer, Jeremy: SenSay: A Context-Aware Mobile Phone. In: *Proceedings of the 16th International Symposium on Wearable Computers*, 2012

[139] Smith, Julius O.: *Spectral Audio Signal Processing*. Stanford : W3K Publishing, 2011

[140] Sokolova, Marina ; Lapalme, Guy: A Systematic Analysis of Performance Measures for Classification Tasks. In: *Information Processing & Management* 45 (2009), Nr. 4

[141] Srivastava, Smriti ; Bhardwaj, Saurabh ; Bhandari, Abhishek ; Gupta, Krit ; Bahl, Hitesh ; Gupta, J.R.P.: Wavelet Packet Based Mel Frequency Cepstral Features for Text Independent Speaker Identification. In: *Intelligent Informatics*. 2013

[142] Stager, M. ; Lukowicz, P. ; Troster, G.: Implementation and Evaluation of a Low-power Sound-based User Activity Recognition System. In: *Proceedings of the Eighth International Symposium on Wearable Computers*, 2004

[143] Stein, Sebastian ; McKenna, Stephen J.: Combining Embedded Accelerometers with Computer Vision for Recognizing Food Preparation Activities. In: *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2013

[144] Steinmetz, Ralf ; Nahrstedt, Klara: *Multimedia Applications*. Berlin, New York : Springer, 2004

[145] Stiefmeier, Thomas ; Roggen, Daniel ; Ogris, Georg ; Lukowicz, Paul ; Tröster, Gerhard: Wearable Activity Tracking in Car Manufacturing. In: *IEEE Pervasive Computing* 7 (2008), Nr. 2

[146] Sun, Xiao ; Lu, Zongqing ; Hu, Wenjie ; Cao, Guohong: SymDetector: Detecting Sound-related Respiratory Symptoms Using Smartphones. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[147] Tan, Wai-Tian ; Baker, Mary ; Lee, Bowon ; Samadani, Ramin: The Sound of Silence. In: *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2013

[148] Tsai, Ming-Chang ; Chou, Fu-Chiang ; Kao, Yih-Feng ; Yang, Kai-Cheng ; Chen, Mike: Polite Ringer II: a Ringtone Interaction System Using Sensor Fusion. In: *Proceedings of the 13th International Conference on Ubiquitous Computing*, 2011

[149] Turner, Liam D. ; Allen, Stuart M. ; Whitaker, Roger M.: Interruptibility Prediction for Ubiquitous Systems: Conventions and New Directions from a Growing Field. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[150] Tzanetakis, George ; Cook, Perry: Musical Genre Classification of Audio Signals. In: *IEEE Transactions on Speech and Audio Processing* 10 (2002), Nr. 5

[151] Vergin, Rivarol ; O'shaughnessy, Douglas ; Farhat, Azarshid: Generalized Mel Frequency Cepstral Coefficients for Large-Vocabulary Speaker-Independent Continuous-Speech Recognition. In: *IEEE Transactions on Speech and Audio Processing* 7 (1999), Nr. 5

[152] Vu, Long ; Do, Quang ; Nahrstedt, Klara: Jyotish: Constructive Approach for Context Predictions of People Movement from Joint Wifi/Bluetooth Trace. In: *Pervasive and Mobile Computing* 7 (2011), Nr. 6

[153] Vu, Long ; Nahrstedt, Klara ; Retika, Samuel ; Gupta, Indranil: Joint Bluetooth/Wifi Scanning Framework for Characterizing and Leveraging People Movement in University Campus. In: *Proceedings of the 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, 2010

[154] Walt, Stéfan van der ; Colbert, S. C. ; Varoquaux, Gaël: The NumPy Array: A Structure for Efficient Numerical Computation. In: *Computing in Science and Engineering* 13 (2011), Nr. 2

[155] Wang, He ; Sen, Souvik ; Elgohary, Ahmed ; Farid, Moustafa ; Youssef, Moustafa ; Choudhury, Romit R.: No Need to War-drive: Unsupervised Indoor Localization. In: *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2012

[156] Wang, Rui ; Chen, Fanglin ; Chen, Zhenyu ; Li, Tianxing ; Harari, Gabriella ; Tignor, Stefanie ; Zhou, Xia ; Ben-Zeev, Dror ; Campbell, Andrew T.: StudentLife: Assessing Mental Health, Academic Performance and Behavioral Trends of College Students Using Smartphones. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2014

[157] Ward, J.A. ; Lukowicz, P. ; Troster, G. ; Starner, T.E.: Activity Recognition of Assembly Tasks Using Body-Worn Microphones and Accelerometers. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), Nr. 10

[158] Wiese, Jason ; Saponas, T. S. ; Brush, A.J. B.: Phoneprioception: Enabling Mobile Phones to Infer Where They Are Kept. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2013

[159] Wu, Wanmin ; Dasgupta, Sanjoy ; Ramirez, E. E. ; Peterson, Carlyn ; Norman, J. G.: Classification Accuracies of Physical Activities Using Smartphone Motion Sensors. In: *Journal of Medical Internet Research* 14 (2012), Nr. 5

[160] Wu, Wei ; Ng, Wee S. ; Krishnaswamy, Shonali ; Sinha, Abhijat: To Taxi or Not to Taxi? - Enabling Personalised and Real-Time Transportation Decisions for Mobile Users. In: *Proceedings of the 2012 IEEE 13th International Conference on Mobile Data Management*, 2012

[161] Xu, Qiang ; Zheng, Rong ; Hranilovic, Steve: IDyLL: Indoor Localization Using Inertial and Light Sensors on Smartphones. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2015

[162] Yuan, Jing ; Zheng, Yu ; Zhang, Liuhang ; Xie, XIng ; Sun, Guangzhong: Where to Find My Next Passenger. In: *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp)*, 2011

[163] Zhou, Pengfei ; Zheng, Yuanqing ; Li, Zhenjiang ; Li, Mo ; Shen, Guobin: IODetector: A Generic Service for Indoor Outdoor Detection. In: *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SenSys)*, 2012

[164] Zijlstra, Fred R. ; Roe, Robert A. ; Leonora, Anna B. ; Krediet, Irene: Temporal Factors in Mental Work: Effects of Interrupted Activities. In: *Journal of Occupational and Organizational Psychology* 72 (1999), Nr. 2

A

APPENDIX

A.1 QUESTIONNAIRE REGARDING MANUAL PHONE MODE ADAPTATION

In what follows we present the complete questionnaire that participants in our user study had to fill in. The user study was accessible online, each of the five sections of the questionnaire being shown on individual pages. Several questions had free fields for users to enter their options in case these were not included in the predefined choices.

A.1.1 *Start Page*

Welcome to our survey! We are looking into how often users adapt their phone mode and how often they have to deal with disturbances/missed phonecalls when they forget to do so.

1. Gender
Please select your gender

- Female
- Male

2. Age
Please select your age

- Options between 14 and 100 years old

3. Occupation
Please select your current professional activity. If nothing fits, please select "other" and fill in the box below your occupation.

- Student
- Researcher
- Office worker
- Manual worker
- Stay at home mom/dad
- Unemployed
- Other:

A.1.2 *Phone Settings That You Adapt Manually*

1. What phone settings do you adapt manually? Please check all that apply.
Phone mode = settings regarding how the phone signals when a phonecall, message or notification comes in, or when an alarm goes off. These settings refer to the ringtones, notification messages and vibration motor.

- Phone mode (silent, ring, vibrate, only light, normal, outdoors)
- Ringtone/notification sounds volume
- Screen brightness
- Presence information on communication apps (Skype, Whatsapp, Telegram, etc.)
- Location Services (on/off)
- Flight Mode (on/off)
- Other:

A.1.3 *When and Why You Adapt Your Phone Mode*

1. For adapting your phone mode, how relevant are the following criteria? [1-5 scale, from "not relevant" to "very relevant"]

- The place where you hold you phone (bag, pocket, hand, etc.)
- The activity you are performing at the moment (being in a meeting, having a break, commuting to/from work, watching a movie at the cinema etc.)
- Environment conditions (loudness, movement, etc.)
- Other:

2. How do you usually carry (or store) your phone? Please estimate how often you use each of the positions listed below. If you use other positions, please select one or more of the "other" options and fill in below what those positions are. [1-5 scale, from "never" to "all the time"]

- In a pocket
- In a bag
- In my hand
- On a desk/table
- Other (1):
- Other (2):
- Other (3):

3. For setting your phone mode to "only lights" mode, how relevant are the following activities? [1-5 scale, from "not relevant" to "very relevant"]

"Only lights" mode means that just the screen lights up when a new phonecall/message comes, no sound or vibration.

- Meeting or discussion
- Attending or giving presentation
- Phonecall
- Working on your computer (in your office)
- Break
- "Physical" activity around the office (e.g., add paper or get printer unclogged, do kitchen duty, etc.)
- Commuting
- Other:

4. For setting your phone to silent mode, how relevant are the following activities? [1-5 scale, from "not relevant" to "very relevant"]

Silent mode means that the phone signals new phonecalls/messages through vibrations and no sound.

- Meeting or discussion
- Attending or giving presentation
- Phonecall
- Working on your computer (in your office)
- Break
- "Physical" activity around the office (e.g., add paper or get printer unclogged, do kitchen duty, etc.)
- Commuting
- Other:

5. For setting your phone to normal mode, how relevant are the following activities? [1-5 scale, from "not relevant" to "very relevant"]

Normal phone mode means that new phonecalls/messages are announced through sounds at normal volume (and no vibrations).

- Meeting or discussion
- Attending or giving presentation
- Phonecall

- Working on your computer (in your office)
- Break
- "Physical" activity around the office (e.g., add paper or get printer unclogged, do kitchen duty, etc.)
- Commuting
- Other:

6. For setting your phone to loud mode, how relevant are the following activities? [1-5 scale, from "not relevant" to "very relevant"]

Loud mode means that incoming phonecalls/messages are signaled through loud sounds and vibrations at the same time.

- Meeting or discussion
- Attending or giving presentation
- Phonecall
- Working on your computer (in your office)
- Break
- "Physical" activity around the office (e.g., add paper or get printer unclogged, do kitchen duty, etc.)
- Commuting
- Other:

A.1.4 *How Often You Adapt Your Phone Mode*

For each question there was a seven-value scale, each value marked respectively: never, less than once per week, once per week once every few days, 1-2 times per day, 3-5 times per day, more than 5 times per day.

1. How often do you adapt your phone mode?
2. How often do you happen to forget to adapt the phone mode and it leads to disturbance/missed calls?
3. How often do people you interact with/surrounding people happen to forget to adapt the phone mode and this leads to disturbance?
4. How often do you miss phonecalls because your phone is on silent/in the wrong mode?

A.1.5 *Opinion About Automated Solutions*

1. Could you imagine using an app that uses smartphone sensors to determine your activity and phone position, and based on this adapts the phone mode?

If you pick the third option, please add a short explanation of what conditions you'd require for using the app. If you pick the forth or fifth option, please explain briefly why you wouldn't use such an application.

- Yes, definitely
- Yes, as long as it is done in a privacy-conscious manner: all data is processed on the phone and nothing is transferred from the phone, no personal information is collected, the application functions in a totally transparent manner
- Yes, under some other condition(s)
- Maybe/Undecided
- No

2. Could you imagine using an app that uses smartphone sensors to determine your activity and phone position, and based on this notifies you to change your phone mode?

If you pick the third option, please add a short explanation of what conditions you'd require for using the app. If you pick the forth or fifth option, please explain briefly why you wouldn't use such an application.

- Yes, definitely
- Yes, as long as it is done in a privacy-conscious manner: all data is processed on the phone and nothing is transferred from the phone, no personal information is collected, the application functions in a totally transparent manner
- Yes, under some other condition(s)
- Maybe/Undecided
- No

B

AUTHOR'S PUBLICATIONS

B.1 MAIN PUBLICATIONS

1. Irina Diaconița, Andreas Reinhardt, Delphine Christin, and Christoph Rensing. Inferring Smartphone Positions based on Collecting the Environment's Response to Vibration Motor Actuation. In *Proceedings of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM) Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet)*, 2015.

2. Irina Diaconița, Andreas Reinhardt, Delphine Christin, and Christoph Rensing. Bleep Bleep! Determining Smartphone Locations by Opportunistically Recording Notification Sounds. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, 2014.

3. Irina Diaconița, Christoph Rensing, and Stephan Tittel. Getting the Information You Need, When You Need It: A Context-aware Q&A System for Collaborative Learning. In *Proceedings of the 9th European Conference on Technology Enhanced Learning*, 2014.

4. Irina Diaconița, Andreas Reinhardt, Frank Englert, Delphine Christin, and Ralf Steinmetz. Do You Hear What I Hear? Using Acoustic Probing to Detect Smartphone Locations. In *Proceedings of the 12th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2014.

5. Irina Diaconița, Christoph Rensing, and Stephan Tittel. Context-aware Question and Answering for Community-based Learning. In Proceedings of "Die 11. e-Learning Fachtagung Informatik", 2013.

B.2 CO-AUTHORED PUBLICATIONS

1. Lena Després, Christoph Rensing, and Irina Diaconița. Konzeption, Durchführung und Ergebnisse einer Usability-Studie zu einem Q&A System für mobiles Lernen von Servicetechnikern. In *Proceedings of "Die 12. e-Learning Fachtagung Informatik"*, 2014

2. Christoph Rensing and Irina Diaconița. A Q&A system considering employees' willingness to help colleagues and to look for help in different workplace-related situations. In *Proceedings 14th International Conference on Advanced Learning Technologies (ICALT)*, 2014.

3. Frank Englert, Irina Diaconița, Andreas Reinhardt, Alaa Alhamoud, and Ralf Steinmetz. Reduce the Number of Sensors: Sensing Acoustic Emissions to Estimate Appliance Energy Usage. In *Proceedings of the 5th ACM Workshop On Embedded Systems For Energy-Efficient Buildings (BuildSys)*, 2013.

C

CURRICULUM VITÆ

PERSONAL DETAILS

Name	Irina Diaconița
Date of Birth	2 January 1988
Place of Birth	Iași, Romania
Nationality	Romanian

EDUCATION

since 10/2012	**Doctoral researcher**, KOM – Multimedia Communications Lab, Technische Universität Darmstadt, Germany
10/2010-06/2012	**Master of Science in Computer Science**, Technical University of Iași, Romania
10/2006–07/2010	**Bachelor of Science in Computer Science**, Technical University of Iași, Romania
09/2002–07/2006	**Highschool** Costache Negruzzi National College, Iași, Romania

WORK EXPERIENCE

since 09/2017	**Data Scientist**, Main Incubator GmbH (Subsidiary of Commerzbank AG) Frankfurt, Germany
10/2012–02/2017	**Research Assistant**, KOM – Multimedia Communications Lab, Technische Universität Darmstadt, Germany

SCIENTIFIC ACTIVITIES

2017	Reviewer for Pervasive and Mobile Computing Journal
2016	Reviewer for Multimedia Systems Journal

D

SUPERVISED STUDENT THESES

D.1 MASTER THESES

1. KOM-M-0528. Manish Agarwal. *A Distance and Acceleration-based Approach for Determining User Groups in a Classroom*. Master Thesis, Technische Universität Darmstadt, October 2015.

D.2 BACHELOR THESES

1. KOM-B-0563. Daniel Nicolas May. *Learning Diary: Quantified Self Meets Learning*. Bachelor Thesis, Technische Universität Darmstadt, November 2016.
2. KOM-B-0524. Manuel Fernandez Sánchez. *An Audio-based Approach to Determine the Distance Between Smartphones*. Bachelor Thesis, Technische Universität Darmstadt, July 2015.
3. KOM-B-0511. Tobias Michels. *Audio-based System for Detecting User Activity and Co-Participating Peers*. Bachelor Thesis, Technische Universität Darmstadt, November 2014.

E

ERKLÄRUNG LAUT §9 DER PROMOTIONSORDNUNG

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe.

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 4. Dezember 2017

Irina Diaconița